«به نام خدا»

نام: فاطمه زهرا بخشنده استاد: دکتر بهروز مینایی

شماره دانشجویی: 98522157

گزارش تمرین ششم امتیازی:

در این تمرین، میخواهیم یک شبکه ی MLP را آموزش دهیم تا بتواند انواع لباس ها را از یکدیگر تشخیص دهد. برای این کار از مجموعه داده های Fasion Mnist و از کتابخانه ی Keras برای پیاده سازی شبکه ی MLP خود استفاده می کنیم.

ابتدا بخش اول سوال را پیاده سازی میکنیم، سپس به بخش دوم، یعنی پاسخ به قسمت های (آ) تا (ح) می پردازیم.

بخش اول:

ابتدا کتابخانه های مورد نیاز را import میکنیم:

```
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import numpy as np
import random as rnd

from keras.datasets import  fashion_mnist
```

سپس عکس هارا از مجموعه داده Fasion Mnist لود میکنیم:

```
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

Seed ها را با توجه به خواسته سوال تنظیم میکنیم:

```
np.random.seed(0)
rnd.seed(0)
tf.random.set_seed(0)
```

و داده های train_image و test_image را به حالت نرمال می بریم:

```
train_images = train_images / 255
test_images = test_images / 255
```

حالا برای شبکه MLP خود، مقادیر داده شده در سوال را برای پارامترها تنظیم می کنیم. می خواهیم بهترین تعداد نورون های لایه مخفی را از بین مقادیر (256, 128, 64, 32, 16) پیدا کنیم. پس هربار تعداد نورون های لایه مخفی مدل خود را با هرکدام از این داده ها مقداردهی می کنیم و نمودار loss و accuracy را برای هرکدام بررسی میکنیم.

در حالت کلی باید مدلی مانند زیر تعریف کنیم، و هر بار مقدار units را در لایه مخفی آن عوض کنیم:

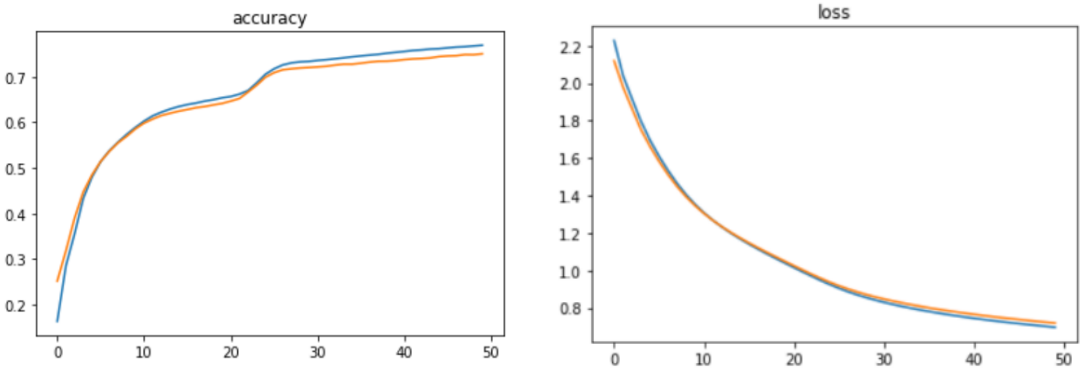
```
model = keras.models.Sequential(
     name='MyModel', layers=[
     keras.layers.Input(shape=(28, 28)),
     keras.layers.Flatten(),
     keras.layers.Dense(units=256, activation=keras.activations.relu),
     keras.layers.Dense(units=10, activation='softmax')
    ]
)

model.compile(
    loss=keras.losses.sparse_categorical_crossentropy,
    optimizer=keras.optimizers.SGD(learning_rate=0.001),
    metrics=['accuracy']
    )
```

و به صورت زیر فرایند آموزش را برای مدل خود انجام می دهیم:

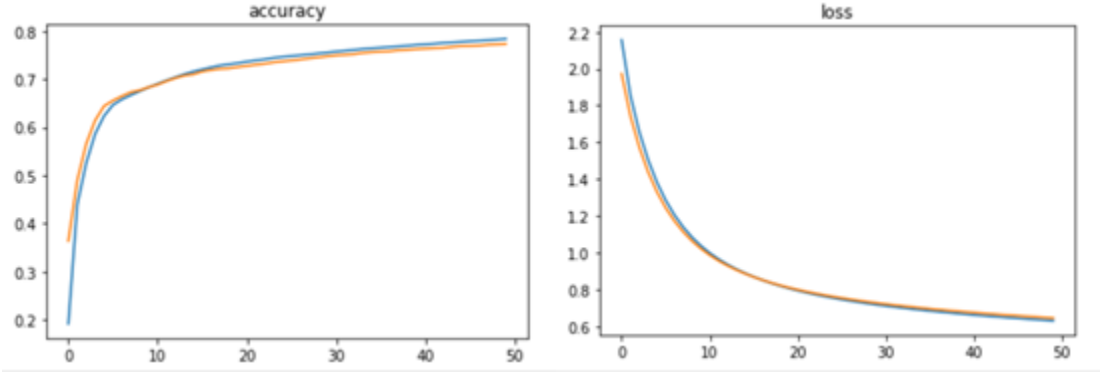
```
result = model.fit(
          train_images,
          train_labels,
          batch_size=256,
          epochs=50,
          validation_data=(test_images, test_labels)
          )
```

حالا برای هر کدام از مقادیر 16 تا 256 برای تعداد نورون ها، این مراحل را تکرار کرده و نتایج آن ها را باهم مقایسه میکنیم.

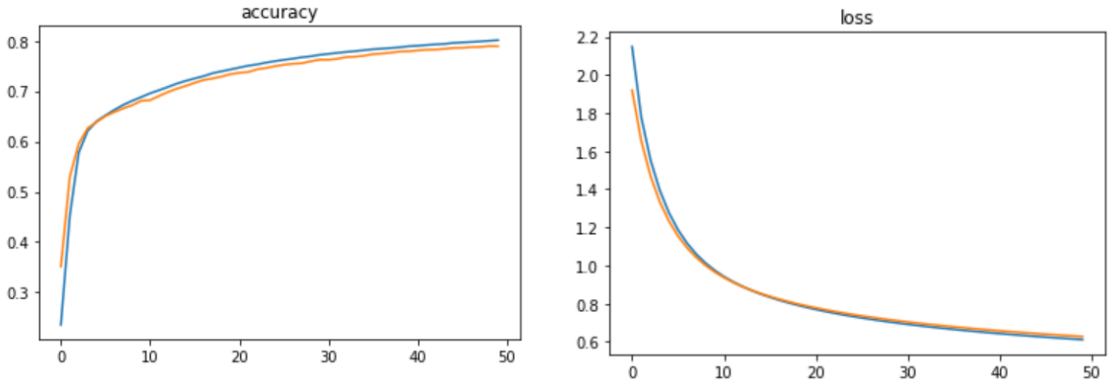
- مدل با تعداد نورون های لایه مخفی 16



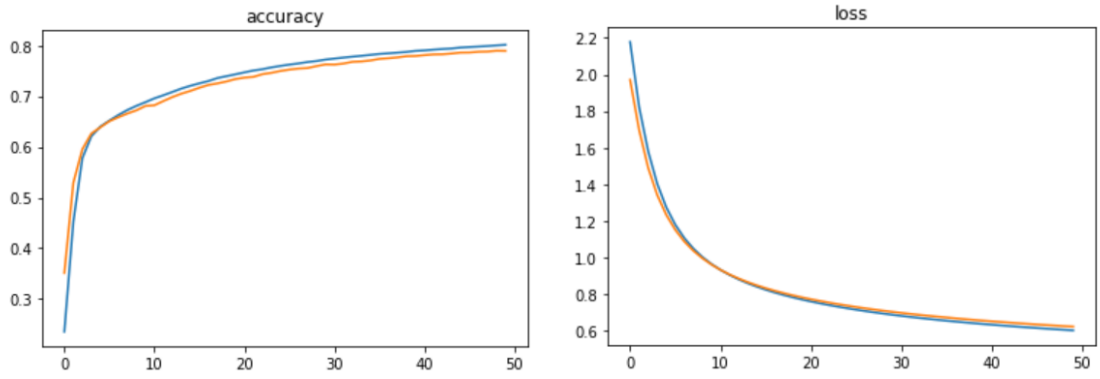
- مدل با تعداد نورون های لایه مخفی 32



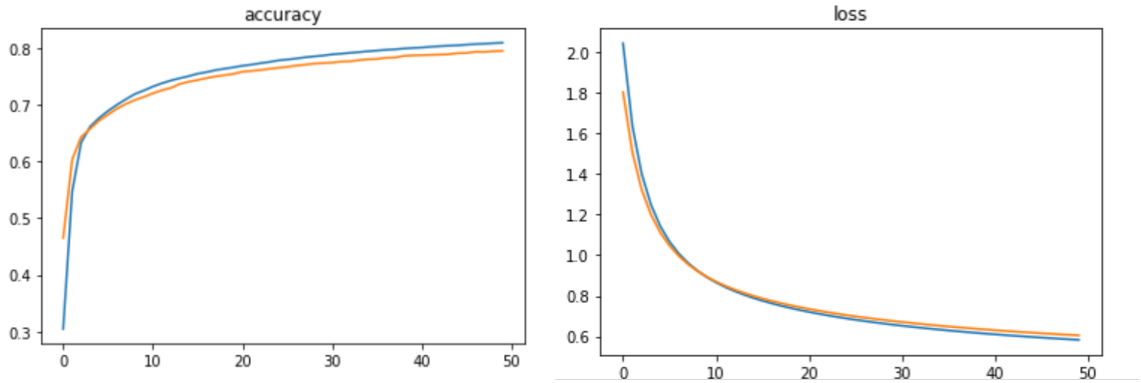
- مدل با تعداد نورون های لایه مخفی 64



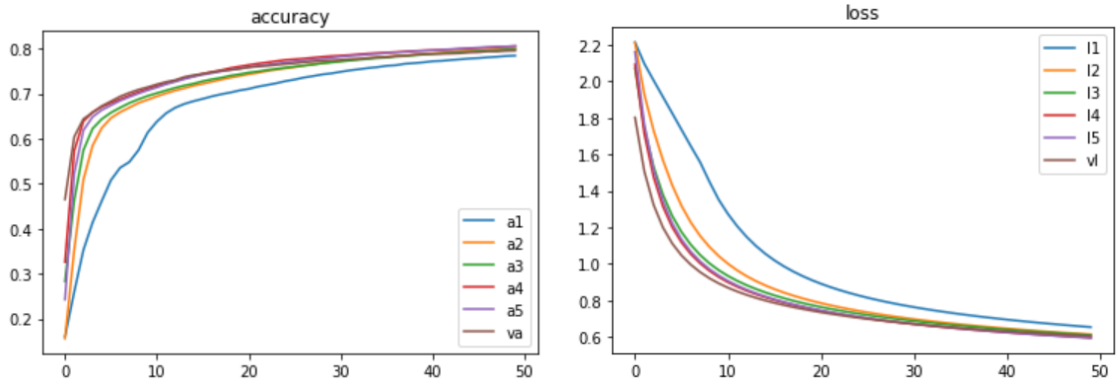
- مدل با تعداد نورون های لایه مخفی 128



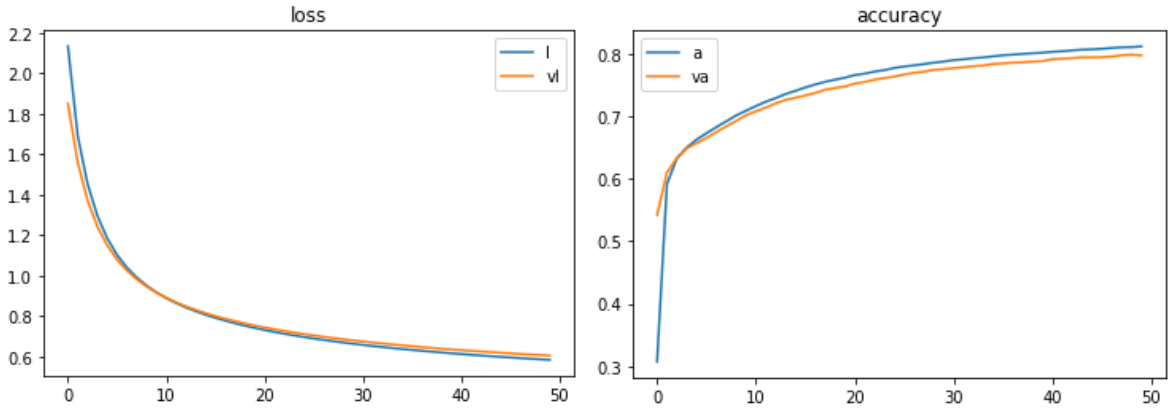
- مدل با تعداد نورون های لایه مخفی 256



در آخر یک بار، پنج مدل با تعداد لایه های مخفی گوناگون داده شده در سوال، ساختم و پس از آموزش تک تک آن ها، همه آن هارا باهم در یک نمودار مقایسه کردم.



با مقایسه نتایج و نمودار loss و accuracy در می یابیم، بیشترین میزان accuracy و کمترین مقدار loss مربوط به مدل با تعداد نورون های مخفی 256 است، در واقع هرچه تعداد نورون های لایه مخفی بیشتر شد، دقت نیز بیشتر شد. پس برای مدل خود از units=256 استفاده میکنیم. آن را آموزش می دهیم که کد آن را قبلا توضیح دادم. و نمودار آن نیز به صورت زیر شد:



حالا دقت مدل را روی داده های تست، evaluate می کنیم:

```
test_loss, test_accuracy = model.evaluate(test_images,  test_labels)
print('Test accuracy:', test_accuracy, ', Test loss:', test_loss)
```

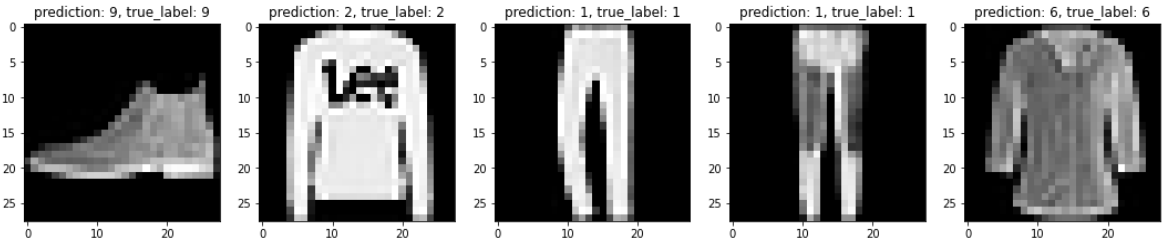
که نتیجه آن به صورت زیر است:

```
Test accuracy: 0.7971000075340271 , Test loss: 0.6091799736022949
```

و برای تست، با مدل آموزش دیده خود روی داده های آزمون، پیش بینی انجام می دهیم و نتیجه آن را برای چند تا از داده ها با label اصلی آن ها مقایسه می کنیم.

```
test = test_images * 255
for i in range(0, 15):
  plt.subplot(3, 5, i + 1)
  plt.imshow(test[i], cmap='gray')
  plt.title(f'prediction: {my_predicts[i]}, true_label: {test_labels[i]}')
plt.subplots_adjust(right=2.5, top=2.3)
plt.show()
```

چند نمونه از نتیجه را اینجا میبینیم:



بخش دوم:

برای حل این بخش تابع model trainer را مینویسیم تا به ازای optimizer و hidden_units و ... که از ورودی میگیرد همین کار هارا تکرار کند. از آن برای پاسخ به سوالات باقی مانده استفاده میکنیم.

آ)

این دیتاست شامل تصاویر grayscale با ابعاد 28 x 28 پیکسل میباشد که شامل 60000 داده آموزشی و 10000 داده آزمون است.

```
print(train_images.shape)
print(train_labels.shape)
print(test_images.shape)
print(test_labels.shape)
```

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

هر یک از تصاویر این دیتاست به یکی از 10 گونه لباس تعلق دارد:

ب)

دیتاست اصلی دارای 60000 داده آموزشی و 10000 داده تست است یعنی نسبت 6 به 1 دارند. و تعداد کل داده ها 70000 تاست که تعداد داده های تست نسبت به تعداد کل داده ها تقریبا 0.14 میشود. مقدار بین 0.16 تا 0.25 برای validation_split میتواند مناسب باشد. یعنی اگر بخواهیم نسبت داده های validation را به کل داده ها حساب کنیم میشود نسبت 0.14 تا 0.2.

در کل بهتر است داده ها به صورتی بین train و validation و test تقسیم شوند که:

1- داده های آموزش بخش عمده را تشکیل داده تا آموزش همه حالات، بهتر و دقیقتر انجام شود.

2- تعداد داده های اعتبارسنجی و تست نیز مقدار معقولی باشد تا حالات زیادی تست و اعتبارسنجی شود.

در ابتدا مقدار 0.2 را انتخاب میکنیم(یعنی نسبت داده های اعتبار سنجی به کل داده ها برابر 0.14 و نسبت داده های اعتبار سنجی به جمع داده های آموزش و اعتبار سنجی برابر 0.2). چون توزیع قبلی داده آموزش و تست را خیلی بر هم نمیزند. همچنین داده های آموزشی به 48000 میرسد که همچنان مقدار مناسبی برای شبکه میباشد.

(یعنی نسبت داده های اعتبار سنجی به کل داده ها برابر 0.17 و نسبت داده های اعتبار سنجی به جمع داده های آموزش و اعتبار سنجی برابر 0.2):

Number of train images: 48000

Number of validation images: 12000

Number of test images: 10000

Train / Validation= 4 , Validation / Total = 12000/70000 = 0.17 , Test / Total = 10000/70000 = 0.14

ج) نتایج را برای تعداد نورونهای مختلف لایه مخفی بررسی می کنیم. (با validation_split=0.2)

Hidden Units	Train	Validation	Test	Analysis
16	loss: 0.6923 accuracy: 0.7632	loss: 0.6888 accuracy: 0.7643	loss: 0.7036 accuracy: 0.7532	loss: Test > Train > Validation accuracy: Validation > Train > Test
32	loss: 0.6753 accuracy: 0.7782	loss: 0.6670 accuracy: 0.7774	loss: 0.6910 accuracy: 0.7651	loss: Test > Train > Validation accuracy: Train > Validation > Test
64	loss: 0.6331 accuracy: 0.7951	loss: 0.6307 accuracy: 0.7932	loss: 0.6510 accuracy: 0.7802	loss: Test > Train > Validation accuracy: Train > Validation > Test
128	loss: 0.6244 accuracy: 0.7979	loss: 0.6176 accuracy: 0.7962	loss: 0.6418 accuracy: 0.7859	loss: Test > Train > Validation accuracy: Train > Validation > Test
256	loss: 0.6222 accuracy: 0.7992	loss: 0.6181 accuracy: 0.7980	loss: 0.6403 accuracy: 0.7881	loss: Test > Train > Validation accuracy: Train > Validation > Test

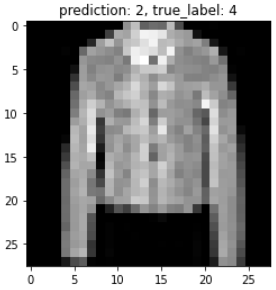
نتیجه به صورت تقریبی:

loss: 16 > 32 > 64 > 128 > 256

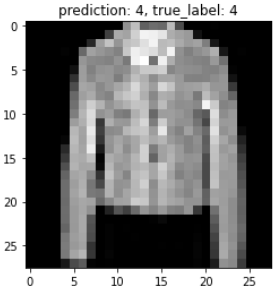
accuracy: 256 > 128> 64 > 32 > 16

تحلیل:

- در حالت 16 نورون لایه مخفی، در حالت آموزش و اعتبارسنجی و آزمون، کمترین دقت و بیشترین خطا را داشتیم. چون اندازه شبکه کوچک بود.
- در حالت 32 نورون لایه مخفی، نسبت به حالت قبلی مشکل کمی حل شده و دیگر مشکل حالت قبلی که دقت اعتبارسنجی بیشتر از آموزش بود را ندارد. اما همچنان می توانیم با بزرگ کردن شبکه دقت را بیشتر کنیم.
- در حالت 64 نورون لایه مخفی، نسبت به حالت قبلی از نظر دقت و خطای شبکه به اعداد خوبی رسیدیم. همچنان می توانیم با بزرگ کردن شبکه دقت را بیشتر کنیم.
- تا به اینجا وقتی روی داده های تست، پیش بینی انجام دادیم و یکی از عکس هارا چک کردیم به صورت زیر اشتباه پیش بینی میکرد:



- در حالت 128 نورون لایه مخفی، از نظر دقت و خطای شبکه کمی پیشرفت کردیم. در این حالت پیش بینی روی داده های تست هم دقیقتر شد و بالاخره عکس قبلی را درست حدس زد:



- در حالت 256 نورون لایه مخفی، دقت و خطای فاز آموزش و آزمون بهتر شده اما دقت و خطای فاز اعتبارسنجی کمی پسرفت کرده است. اگر از آن چشم پوشی کنیم و دقت آموزش و تست را ملاک قرار دهیم، میتوانیم این تعداد نورون هارا بهترین تعداد در نظر بگیریم.
- خطا در داده آزمون از داده های دیگر بیشتر است زیرا تعداد ورودی های آن از داده ارزیابی کمتر است و برخلاف داده آموزش، هیچ آموزشی روی آن صورت نمیگیرد.

د) مقدار Validation Split را 3 بار به مقادیر 0.15 و 0.25 و 0.3 تغییر میدهیم و به ازای تعداد نورون هایی که در صورت سوال بود نتیجه را بررسی میکنیم.

1- validation split = 0.15

Hidden Units	Train	Validation	Test
16	loss: 0.7271 accuracy: 0.7560	loss: 0.7251 accuracy: 0.7564	loss: 0.7419 accuracy: 0.7485
32	loss: 0.6630 accuracy: 0.7829	loss: 0.6594 accuracy: 0.7799	loss: 0.6801 accuracy: 0.7697
64	loss: 0.6232 accuracy: 0.7982	loss: 0.6251 accuracy: 0.7971	loss: 0.6424 accuracy: 0.7838
128	loss: 0.6139 accuracy: 0.8007	loss: 0.6120 accuracy: 0.7987	loss: 0.6328 accuracy: 0.7879
256	loss: 0.6124 accuracy: 0.8014	loss: 0.6109 accuracy: 0.8003	loss: 0.6315 accuracy: 0.7894

2- validation split = 0.25

Hidden Units	Train	Validation	Test
16	loss: 0.7210 accuracy: 0.7610	loss: 0.7165 accuracy: 0.7637	loss: 0.7322 accuracy: 0.7520
32	loss: 0.6888 accuracy: 0.7672	loss: 0.6836 accuracy: 0.7679	loss: 0.7051 accuracy: 0.7550
64	loss: 0.6525 accuracy: 0.7895	loss: 0.6522 accuracy: 0.7855	loss: 0.6716 accuracy: 0.7806
128	loss: 0.6416 accuracy: 0.7904	loss: 0.6400 accuracy: 0.7880	loss: 0.6567 accuracy: 0.7806
256	loss: 0.6226 accuracy: 0.7999	loss: 0.6208 accuracy: 0.7973	loss: 0.6410 accuracy: 0.7881

3- validation split = 0.3

Hidden Units	Train	Validation	Test
16	loss: 0.7365 accuracy: 0.7595	loss: 0.7318 accuracy: 0.7579	loss: 0.7525 accuracy: 0.7473
32	loss: 0.7196 accuracy: 0.7530	loss: 0.7142 accuracy: 0.7538	loss: 0.7334 accuracy: 0.7462
64	loss: 0.6623 accuracy: 0.7831	loss: 0.6590 accuracy: 0.7834	loss: 0.6794 accuracy: 0.7693
128	loss: 0.6477 accuracy: 0.7926	loss: 0.6478 accuracy: 0.7903	loss: 0.6665 accuracy: 0.7820
256	loss: 0.6391 accuracy: 0.7934	loss: 0.6367 accuracy: 0.6367	loss: 0.6583 accuracy: 0.7808

نتیجه:

در هر 3 حالت جدیدی که برای Validation Split داشتیم، مقدار loss و accuracy تقریبا برای تعداد نورون های مختلف به صورت زیر مقایسه میشود:

loss: 16 > 32 > 64 > 128 > 256

accuracy: 256 > 128> 64 > 32 > 16

پس در انتخاب بهترین تعداد نورون های لایه مخفی تغییری ایجاد نشد و میتوان بهترین مقدار آن را تا حدودی مستقل از Validation Split بدست آورد.

دقت فاز آزمون برای تعداد نورون های مختلف و Validation Split های مختلف به صورت زیر است:

- 16 Hidden Units

  test accuracy: VS (0.20) > VS (0.25) > VS (0.15) > VS (0.30)

  test loss: VS (0.30) > VS (0.15) > VS (0.25) > VS (0.20)

- 32 Hidden Units

  test accuracy: VS (0.15) > VS (0.20) > VS (0.25) > VS (0.30)

  test loss: VS (0.30) > VS (0.25) > VS (0.20) > VS (0.15)

- 64 Hidden Units

  test accuracy: VS (0.15) > VS (0.25) > VS (0.20) >= VS (0.30)

  test loss: VS (0.30) > VS (0.25) > VS (0.20) > VS (0.15)

- 128 Hidden Units

  test accuracy: VS (0.15) > VS (0.20) > VS (0.30) > VS (0.25)

  test loss: VS (0.30) > VS (0.25) > VS (0.20) > VS (0.15)

- 256 Hidden Units

  test accuracy: VS (0.15) > VS (0.25) >= VS (0.20) > VS (0.30)

  test loss: VS (0.30) > VS (0.25) > VS (0.20) > VS (0.15)

در حالت های با نورون های بالاتر (128 و مخصوصا 256) نتایج بسیار نزدیک هستند و Validation Split تاثیر کمی دارد.

دقت داده تست در حالت 64 نورون با VS = 0.25 مقدار بسیار خوبی دارد. پس میتوان با انتخاب آن در شبکه کوچکتر نتیجه مطلوب گرفت. همین موضوع باعث میشود مقدار Validation Split را برابر 0.25 در نظر بگیریم.

تاثیر این پارامتر به این گونه است که اگر مقدار آن کم باشد اکثر داده ها به بخش آموزش منتقل میشود و شبکه آموزش بهتری میبیند، اما با ضعیف بودن ارزیابی دقت داده آزمون کاهش مییابد. و اگر این مقدار بیش از اندازه بزرگ باشد شبکه آموزش کمتری میبیند.

۵) قبلا شبکه را با SGD بررسی کرده بودیم. حالا آن را با تغییر optimizer آموزش و ارزیابی میکنیم.
(با validation split = 0.25 و hidden units = 256)

Optimizer	Train	Validation	Test
Adam	loss: 0.0882 accuracy: 0.9693	loss: 0.3742 accuracy: 0.8957	loss: 0.4001 accuracy: 0.8916
RMSprop	loss: 0.0896 accuracy: 0.9680	loss: 0.4173 accuracy: 0.8924	loss: 0.4702 accuracy: 0.8884
Adagrad	loss: 0.5150 accuracy: 0.8314	loss: 0.5208 accuracy: 0.8252	loss: 0.5397 accuracy: 0.8182

پس:

Train loss: Adam < RMSprop < Adagrad < SGD

Val loss: Adam < RMSprop < Adagrad < SGD

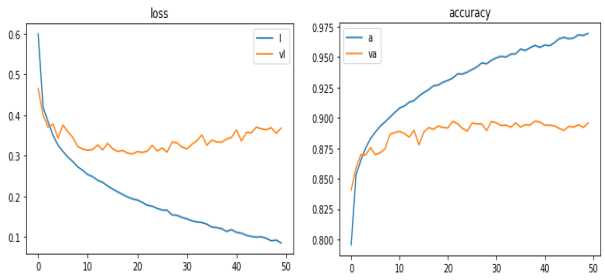
Test loss: Adam < RMSprop < Adagrad < SGD

Train accuracy: Adam > RMSprop > Adagrad > SGD

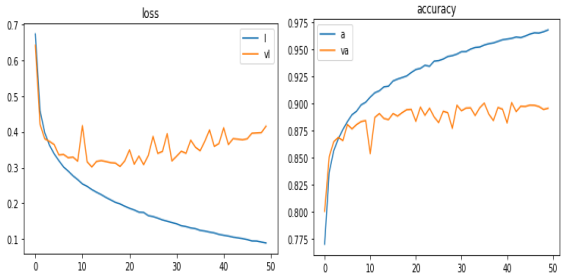
Val accuracy: Adam > RMSprop > Adagrad > SGD

Test accuracy: Adam > RMSprop > Adagrad > SGD

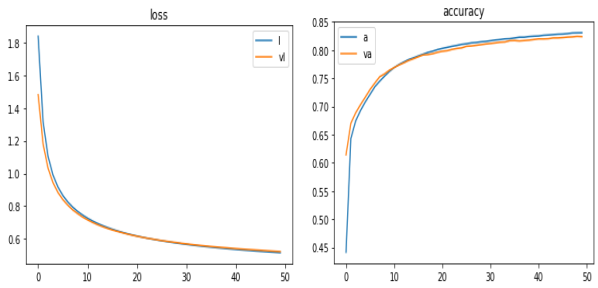
Adam بهترین عملکرد را دارد. زیرا ترکیبی از تمامی حالت هاست و از همه نظر دقت بالاتر و خطای بسیار پایین تری دارد. نمودار همه optimizer ها در زیر آمده است (به اعداد محور عمودی دقت شود):



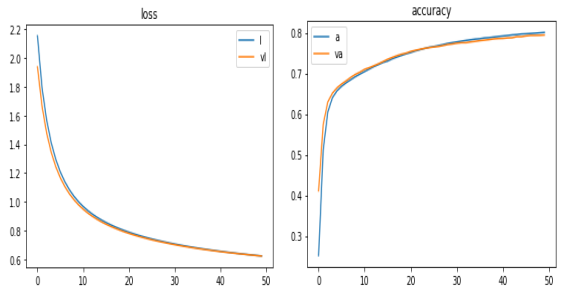
Adam



RMSprop



Adagrad



SGD

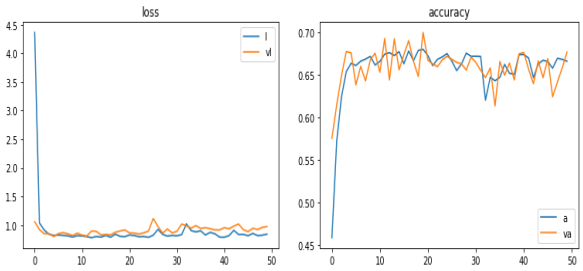
و) شبکه را با تغییر نرخ آموزش، آموزش و ارزیابی میکنیم.
(با validation split = 0.25 و hidden units = 256 و optimizer = Adam)

Learning Rate	Train	Validation	Test
0.1	loss: 0.8417 accuracy: 0.6658	loss: 0.9740 accuracy: 0.6765	loss: 1.0353 accuracy: 0.6769
0.01	loss: 0.1698 accuracy: 0.9369	loss: 0.4851 accuracy: 0.8750	loss: 0.5102 accuracy: 0.8716
0.001	loss: 0.0872 accuracy: 0.9700	loss: 0.3754 accuracy: 0.8965	loss: 0.4119 accuracy: 0.8871
0.0001	loss: 0.2613 accuracy: 0.9089	loss: 0.3296 accuracy: 0.8813	loss: 0.3526 accuracy: 0.8738

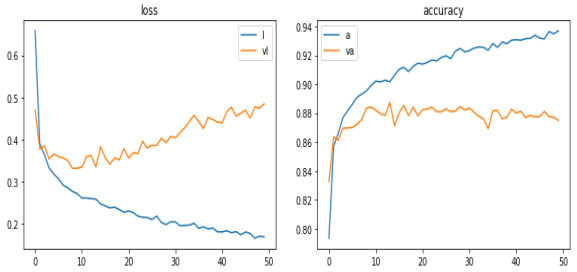
در حالت 0.1 شبکه خطای زیاد و دقت کمی دارد و همگرا نمی شود.

حالت 0.01 بهتر از قبلی است اما هنوز خطای زیادی نسبت به بقیه دارد.

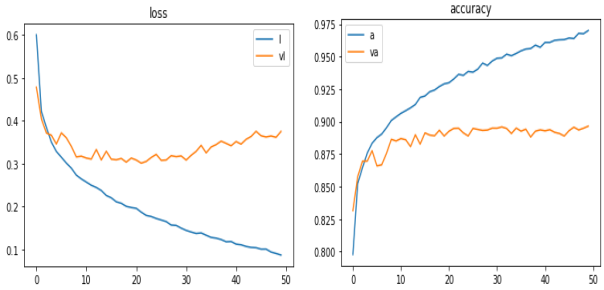
حالت 0.001 از نظر دقت نزدیک به 0.0001 است اما دارای Loss کمتری است و Accuracy آن نیز بیشتر است. البته حالت 0.001 دارای نوسان بیشتری نسبت به 0.0001 است. اما اگر دقت و خطای نهایی را در نظر بگیریم 0.001 بهتر است. اگر برای ما نوسان کم و یکپارچگی مهمتر بود، 0.0001 را در نظر میگرفتیم. در اینجا بهترین نرخ آموزش را مقدار 0.001 نظر می گیریم.



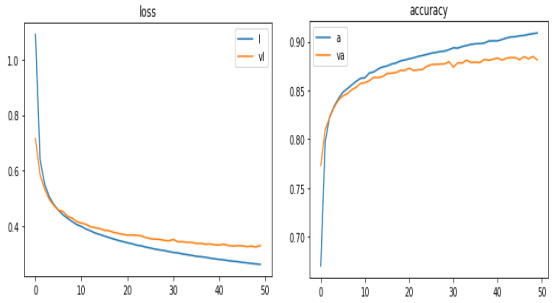
LR = 0.1



LR = 0.01



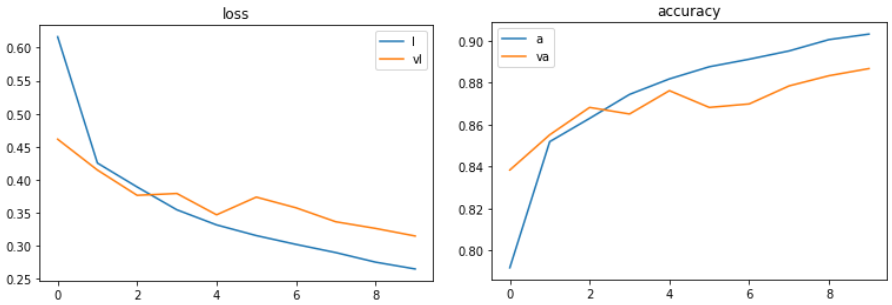
LR = 0.001



LR = 0.0001

ز) شبکه را با Epoch برابر 10 آموزش و ارزشیابی میکنیم. (validation split = 0.25 و hidden units = 256 و optimizer = Adam و learning rate = 0.001)

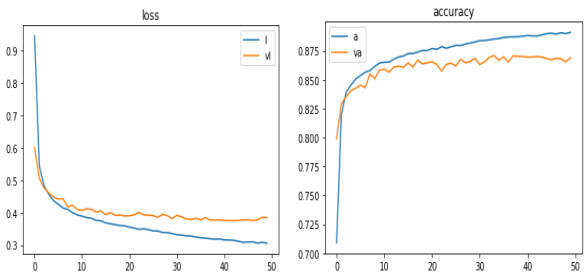
Epochs	Train	Validation	Test
10	loss: 0.2645 accuracy: 0.9032	loss: 0.3145 accuracy: 0.8868	loss: 1.3189 accuracy: 0.4650



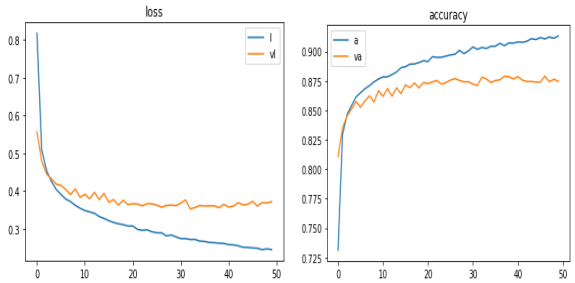
این تعداد epochs برای آموزش خیلی کم است. شبکه همگرا نشده است. مقدار Loss بزرگ است.

ح) این بخش، در قسمت ج به طور کامل تحلیل و تشریح شده است. اما برخی پارامترهای ما تغییر کرده است. (validation split = 0.25 و hidden units = 256 و optimizer = Adam و learning rate = 0.001)

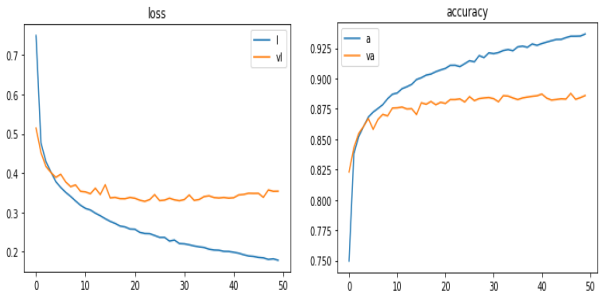
نمودار ها برای تعداد مختلف نورون های لایه مخفی با پارامتر های جدید، به صورت زیر است:



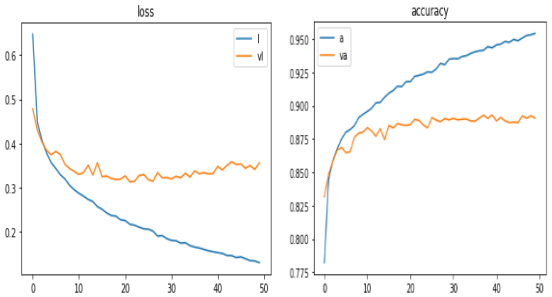
Hidden Units = 16



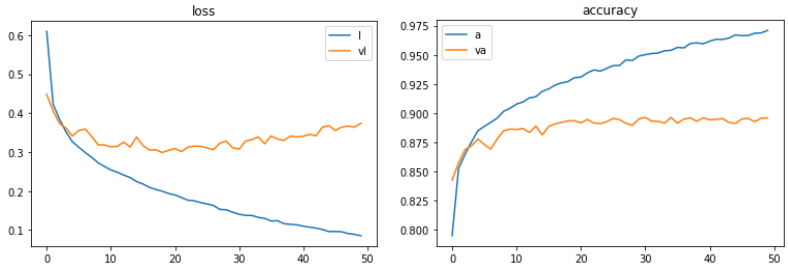
Hidden Units = 32



Hidden Units = 64



Hidden Units = 128



Hidden Units = 256

مشابه قسمت ج:

loss: 16 > 32 > 64 > 128 > 256

accuracy: 256 > 128> 64 > 32 > 16

مشابه توضیحات قسمت ج برای این قسمت نیز صادق است:

- در حالت 16 نورون لایه مخفی، در حالت آموزش و اعتبارسنجی و آزمون، کمترین دقت و بیشترین خطا را داریم. چون اندازه شبکه کوچک است.
- در حالت 32 نورون لایه مخفی، نسبت به حالت قبلی مشکل کمی حل شده اما همچنان می توانیم با بزرگ کردن شبکه دقت را بیشتر کنیم.
- در حالت 64 نورون لایه مخفی، نسبت به حالت قبلی از نظر دقت و خطای شبکه به اعداد خوبی رسیدیم. همچنان می توانیم با بزرگ کردن شبکه دقت را بیشتر کنیم.
- در حالت 128 نورون لایه مخفی، از نظر دقت و خطای شبکه پیشرفت کردیم.
- در حالت 256 نورون لایه مخفی، دقت و خطای فاز آموزش و آزمون بسیار عالی است. این تعداد نورون، بهترین تعداد است.

در آخر، ما با دنبال کردن قسمت (آ) تا (ح) و آزمایش پارامتر های مختلف، بهترین پارامتر هارا برای این شبکه یافتیم:

hidden units = 256

validation split = 0.25

optimizer = Adam

learning rate = 0.001