تمرین سوم یادگیری عمیق

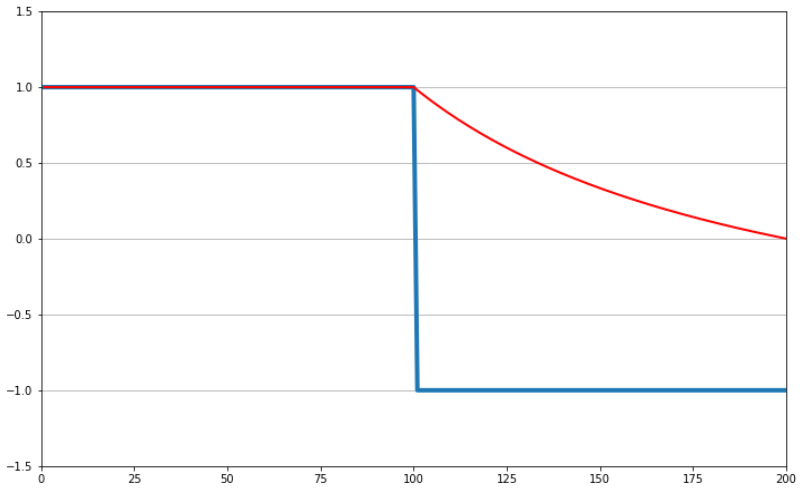
محمدعلی فراهت **97521423**

سوال 1)

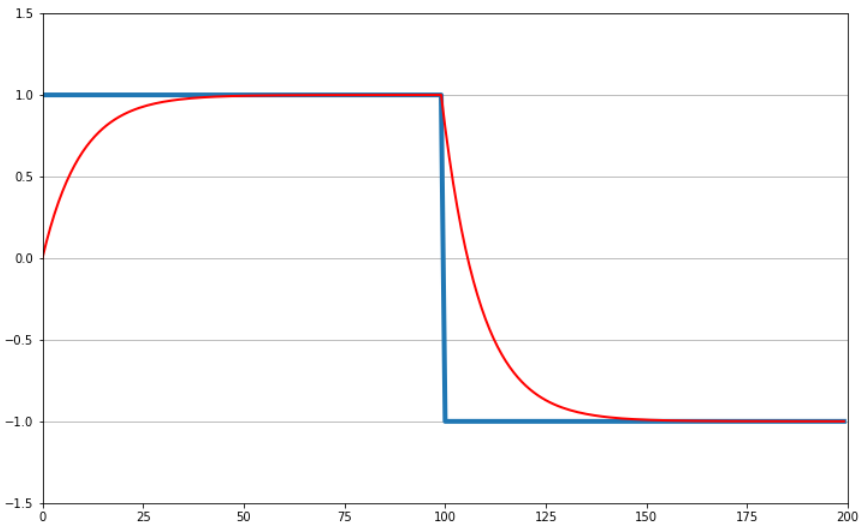
- **الف)** با برسی لینک متوجه شدم که روش بهینه‌سازی Adam معمولا بهتر از بقیه optimizer ها عمل میکند. یعنی در زمان کمتر و با محاسبات کمتر، به نتیجه بهتری میرسد. این optimizer در واقع ترکیبی از AdaGrad و RMSProp است و از مزیت های هر دو دارد استفاده میکند. همانطور که در نمودار هم مشخص است. در تعداد iteration ثابتی ، Adam توانسته training cost را بیشتر کاهش دهد و به دقت بالاتری برسد.
  در سال های اخیر، بعد از پیدایش این روش ، الگوریتم پیشفرض برای بهینه سازی، این الگوریتم است و گاهی از SGD + Momentum هم به انواع جایگزین استفاده میشود.
- **ب)** عملکرد optimizer ها اصولا به کاری که قرار است انجام بدهند بستگی دارد و این یعنی به دیتاست‌ای که قرار است train کنند بستگی دارد. مثلا اگر دیتاست دارای پراکندگی زیادی باشد عملکرد Adam بهتر از بقیه است. یا مثلا اگر در loss function دره های باریکی وجود داشته باشد ، SGD به راحتی آن را پیدا نمیکند و از روی آن رد میشود. به نظر من مهم ترین عوامل عملکرد optimizer ها ، خواص دیتای ورودی آن ها است. و نمیتوان گفت همیشه و در هر موقعیتی یک optimizer خاص بهتر است.

سوال 2)

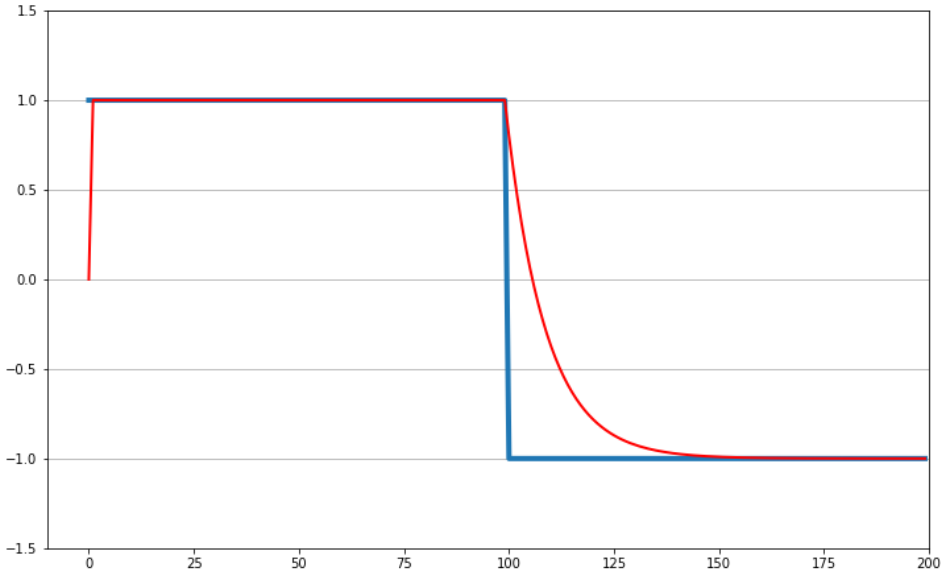
- **الف)** در این بخش این متغیر را (با رنگ آبی) و میانگین عادی آن را (با رنگ قرمز) رسم کردم. میبینیم که تا 100 تکرار اول میانگین مثل خود d است ولی بعد از آن کم کم کاهش پیدا میکند تا به 0 میرسد.



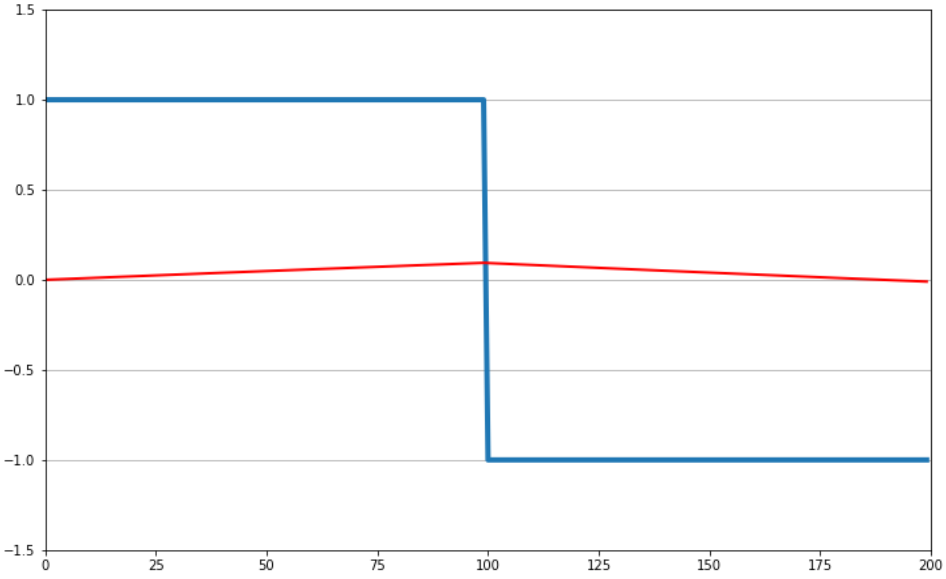
- **ب)** در این بخش با beta = 0.9 و رابطه داده شده (Exponentially Weighted Averages) برای محاسبه میانگین در هر نقطه، خروجی را مانند قبل رسم کردم. با بررسی این نمودار میبینیم که وقتی beta = 0.9 است، تاثیر گذاری نقاط نزدیکتر (حدود 10 نقطه) بیشتر از نقاط دورتر است و میبینیم که نمودار قرمز سریعا از صفر خودش را به 1 رسانده و این یعنی بعد از دورشدن از 0 ، تاثیر آن بر میانگین کمتر میشود. همینطور بعد از منفی شدن نمودار اصلی بعد از 100 ، خیلی سریع تر از قبل خودش را به منفی یک رسانده است. در قسمت د میبینیم که زیاد کردن beta چه تاثیری بر میانگین دارد.



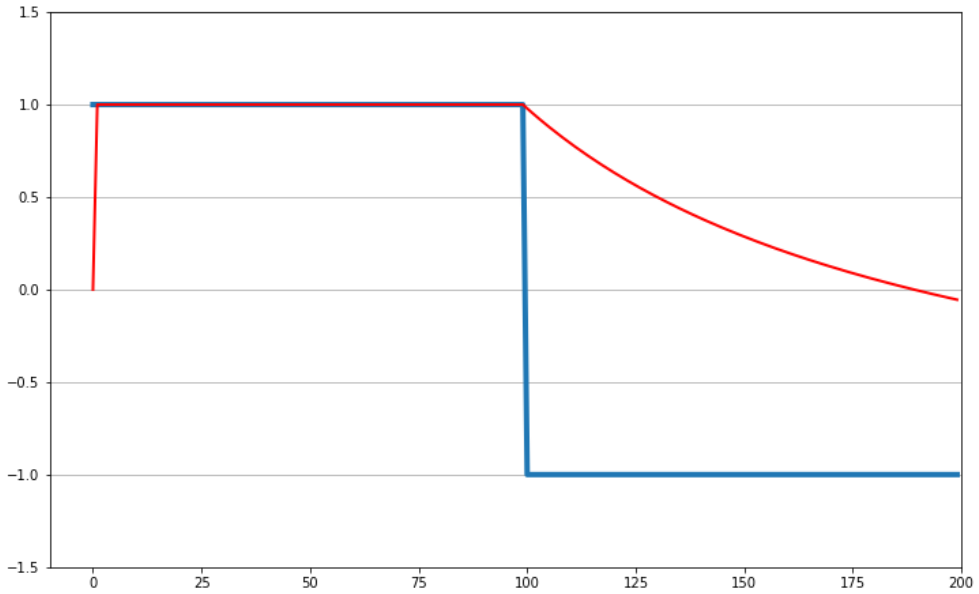
- **ج)** در این بخش مثل بخش ب است ، با این تفاوت که ما از Bias Correction برای بهبود عملکرد این روش (مخصوصا در اوایل نمودار) استفاده میکنیم. اگر به رابطه آن دقت کنیم میبینیم که در iteration های اول ، روی خروجی ما تاثیر بیشتری میگذارد و با زیاد شدن t ، تاثیر آن کمتر و کمتر میشود. همانطور که در نمودار زیر هم مشخص است، در اوایل نمودار میانگین درست حساب میشود و مثل قسمت ب نباید چند iteration بگذرد تا میانگین به 1 برسد.



- **د)** برای m1 ، beta را برابر با 0.999 قرار دادم. با این کار بازه ای که روی میانگین تاثیر میگذارد را زیاد میکنیم (از 10 به 1000). یعنی از پنجره بزرگتری به دیتا نگاه میکنیم. و همانطور که در نمودار زیر مشخص است، تغییرات در نمودار قرمز بسیار کمتر شده‌است.



برای m2 هم، beta را 0.999 قرار دادم و دوباره نمودار را رسم کردم.
با اینکار با اینکه ما پنجره را بزرگ کردیم، ولی چون از Bias Correction استفاده کردیم و در آن هم beta نردیک تر بود به 1 ، تاثیر گذاری آن بیشتر از قبل شد و توانست نمودار را شبیه قسمت قبل دربیاورد، اما با این حال چون پنجره دید ما به دیتا بزرگ بود، نمودار به میانگین عادی خود ، یعنی صفر نزدیک شد.



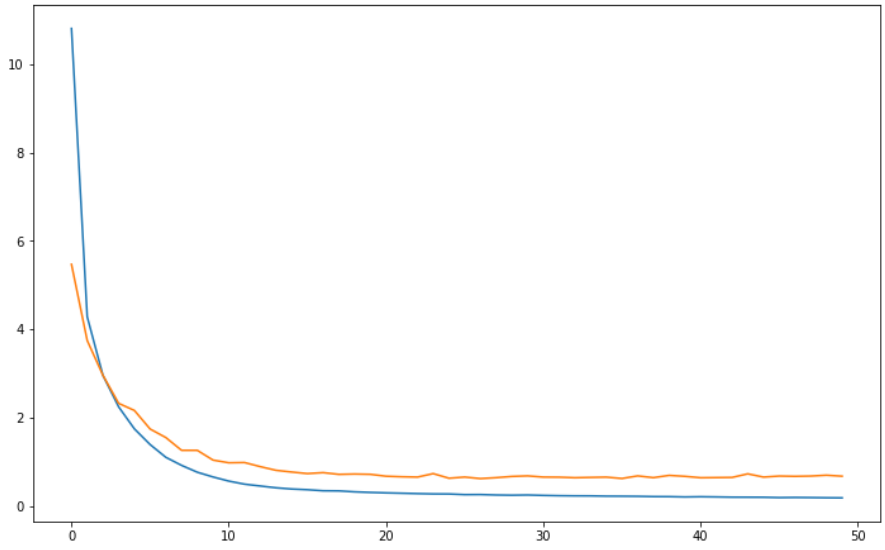
❋ در فایل notebook ، beta را برابر با 0.9 قرار داده ام و اگر قصد تغییر آن را دارید باید در cell دوم و سوم beta را تغییر دهید و به ترتیب اجرا کنید تا خروجی نمایش داده شده شبیه خروجی گزارش شده باشد.

سوال 3) در این بخش یک شبکه MLP را با keras آموزش میدهیم (از اسلاید های آموزش keras استفاده کردم)

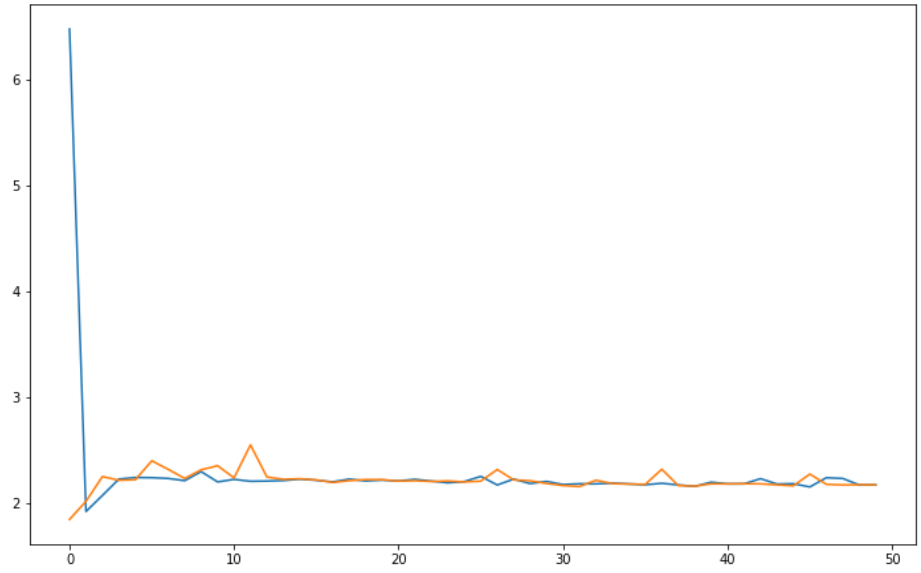
- **الف)** دیتاست Fashion MNIST مجموعه از عکس لباس ها است که در یک 10 طبقه دسته بندی شده‌اند. دارای 60 هزار تصویر برای آموزش، و 10 هزار تصویر برای تست است. هر تصویر سایز 28 * 28 دارد و هر پیکسل grayscale است.
  از ویژگی های آن میتوان به سادگی آموزش روی آن و همچنین درصد بالای یادگیری با اکثر الگوریتم های ماشین لرنینگ اشاره کرد.
- **ب)** من با توجه به جستوجو ای که انجام دادم (لینک منبع) مقدار validation loss را 10 ٪ در نظر گرفتم، چون تعداد داده ها 60 هزار تا بود کم نبود، 10٪ زمان زیادی نمیگرفت.
- **ج)** بعد از اجرا کردن هم 4 حالت برای تعداد نورون لایه میانی، 128 تا نورون بهترین دقت را داشت. برای نمونه، با 16 نورون ، 35٪ دقت داده آموزش و 36.6٪ دقت داده تست بود. همینطور برای 32 و 64 نورون هم افزایش دقت به 57٪ و 78٪ را شاهد بودم. و بهترین گزینه هم 128 نورون بود که دقت آموزش و تست آن به ترتیب 85.3٪ و 84.1٪ بود.
  چون تعداد ورودی ها برای این دیتاست زیاد است ، تعداد نورون های لایه میانی نباید خیلی کم باشند، بنابراین 128 نورون انتخاب منطقی تری است.
- **د)** درصد داده های بخش validation loss را (با توجه به لینک بالا) به 25 درصد و 33 درصد افزایش دادم. در کل دقت 33٪ بهتر از 25٪ بود ، اما بهترین دقت ها برای validation loss = 0.1 بود. با این حال با اینکه درصد دقت تغییر میکرد ولی در همه حالات ، 128 نورون بالاترین دقت را داشت. و این بخاطر دلیل قسمت ج میباشد، تعداد نورون های ورودی زیاد است (28*28) و این نیازمند تعداد بیشتری نورون در در لایه میانی است.
- **ه)** با انتخاب 128 برای تعداد نورون ، 3 نوع optimizer را انتخاب کرده و با validation loss برابر با 0.1 ، آن ها را train کردم. بهترین نتیجه برای Adam بود، که بیش از 83٪ دقت داشت. دوتای دیگر، RMSprop و Adagrad ، نتایج 81 و 80 درصدی گرفتند(آموزش و آزمون بهم خیلی نزدیک بودند) پس با Adam ادامه میدهیم.

- **و)** در این قسمت با learning rate بازی میکنیم. اول مقدار 0.001 را محاسبه میکنیم. دقت تست از حالت قبل خیلی بیشتر شد و به 86.3٪ رسید. و نکته عجیب این است که دقت آموزش (بدون validation) به 93.3٪ هم رسید.

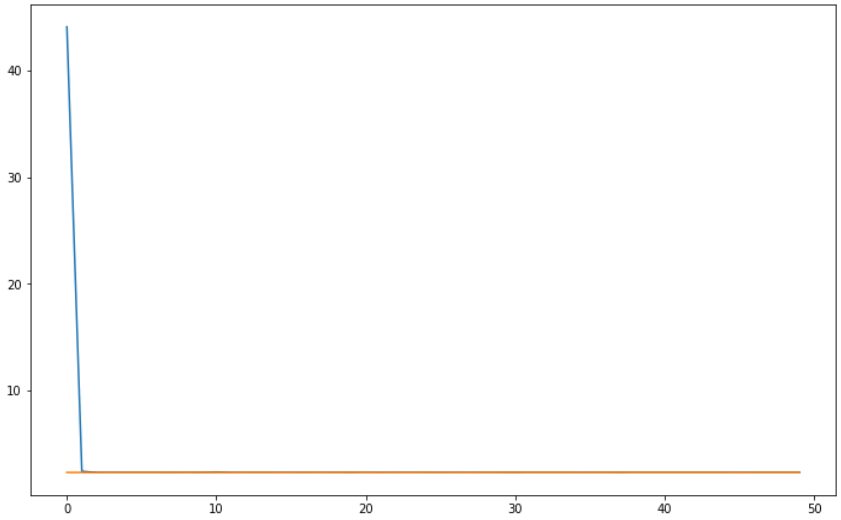
* در نمودارهای زیر خط آبی مقدار loss برای آموزش و نارنجی مقدار validation loss است.



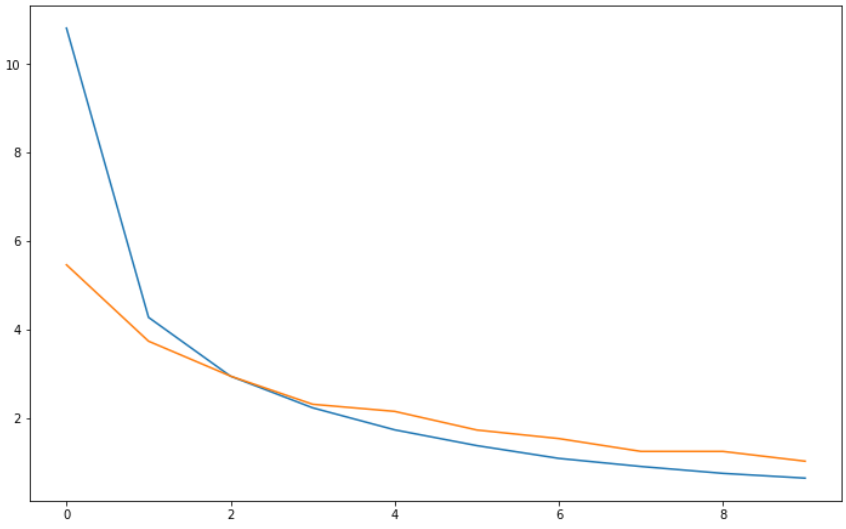
حالا برای learning rate = 0.01 همین کار را تکرار میکنیم ، در اینجا learning rate زیاد است و دقت به شدت پایین آمده و به حدود 15٪ رسیده و امکان همگرایی وجود ندارد. در نمودار همگرایی آن هم ، کاملا مشخص است که learning rate بد بوده.



برای learning rate = 0.1 همانطور که میشد حدس زد، از حالت قبل هم بدتر است و دقت به 10% رسیده و عملا چیزی را یاد نگرفته، چون اگر شانسی هم جواب بدهیم ، دقت همینقدر است.

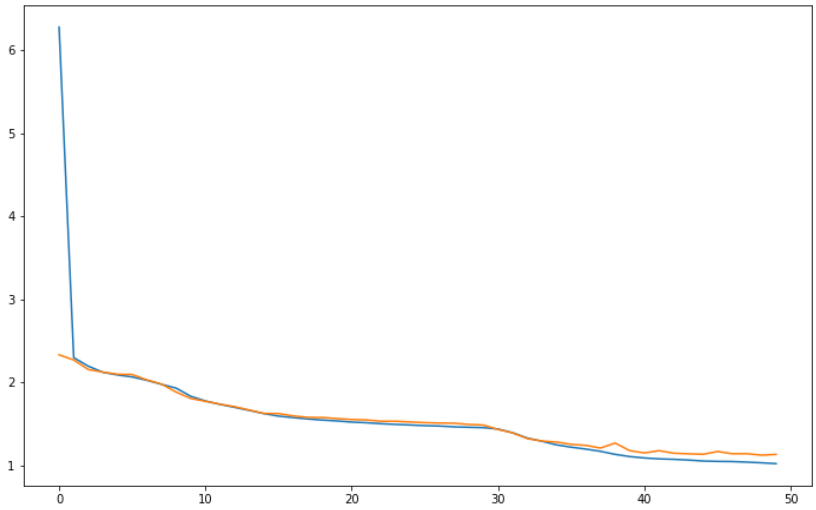


- ز) در این بخش، با learning rate = 0.0001 ادامه دادم و در iteration 10 مدل را train کردم، نمودار آن به شکل زیر است. این مدل همگرا نیست (طبق این تعریف). یعنی اگر ما آموزش را ادامه دهیم ، دقت بیشتر میشود و loss کاهش می‌یابد.

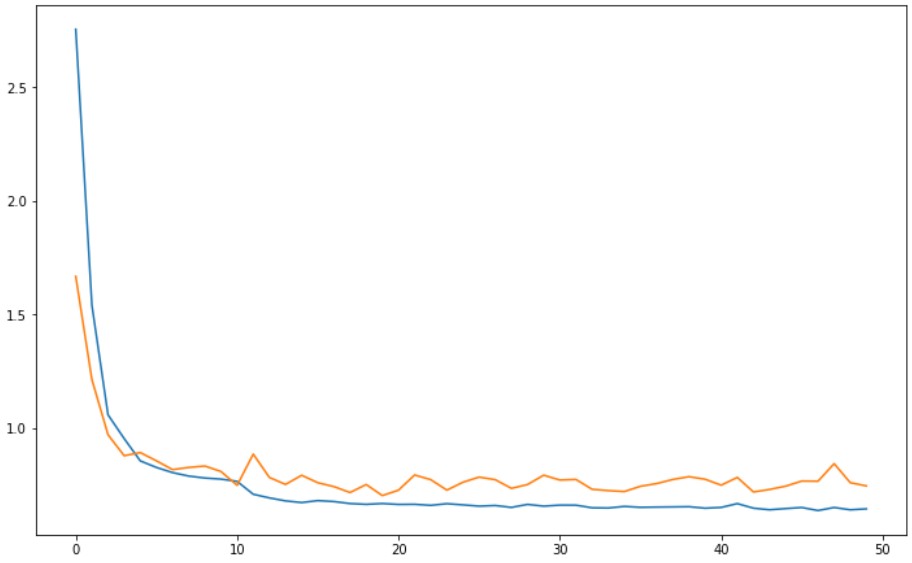


- **ح) از learning rate = 0.0001 استفاده شده.**

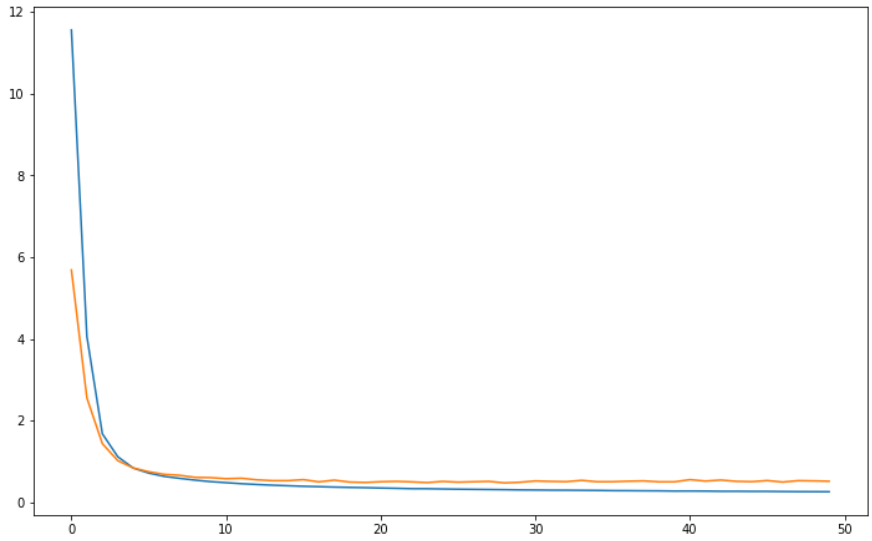
16 نورون : دقت به 56 درصد رسید (هم تست هم آموزش) و حتی میتوانست بهتر هم بشود و هنوز همگرا نشده. و میبینیم که اکثر اوقات نمودار آبی(loss) پایین تر از (validation loss) است.



32 نورون: در این تعداد نورون، دقت افزایش داشته و تا 73٪ رسیده ولی validation loss تغییرات زیادی دارد و با هر iteration تغییر زیادی میکند و تقریبا همیشه بدتر از loss است. با این حال loss تقریبا همگرا شده و نیازی به iteration بیشتر ندارد.



64 نورون : با این تعداد نورون به دقت خوبی رسیدیم، حدود 86٪ ، و loss و validation به خوبی همگرا شده اند و این نشان بخاطر افزایش تعداد نورون ها است.



128 نورون : با افزایش تعداد نورون های لایه میانی به 128 ، دقت پایین تر آمد، حدود 83٪ ، ولی همچنان loss و validation loss همگرا شدند.