گزارش پروژه شبکه عصبی : محمدرضا صادقیان

بخش 1:

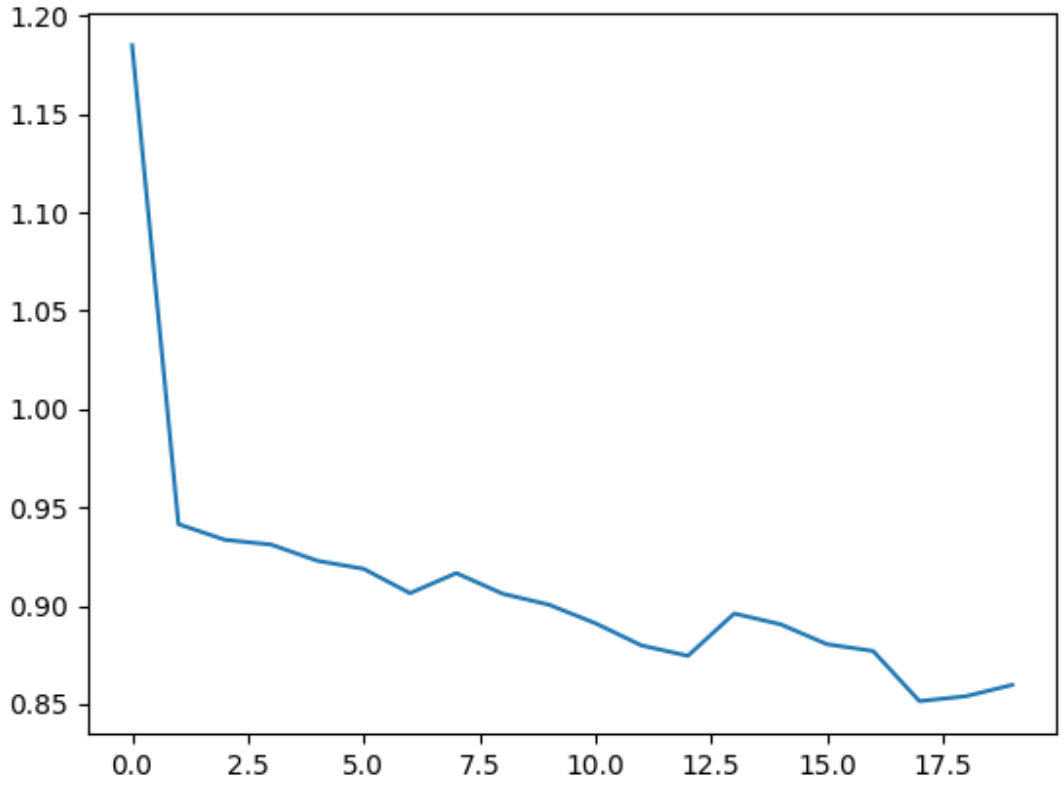
برای بخش 1 کاملا همان کد github را کپی کردم و به ازای چند عدد مختلف چاپ کردم که صرفا چک کنم.

بخش 2:

برای بخش دو صرفا باید با استفاده از وزن و بایاس رندوم خروجی را به دست می آوردیم که من با استفاده از numpy انجام دادم و دقت مدل به صورت میانگین 11.3 بدست آمد.

بخش 3:

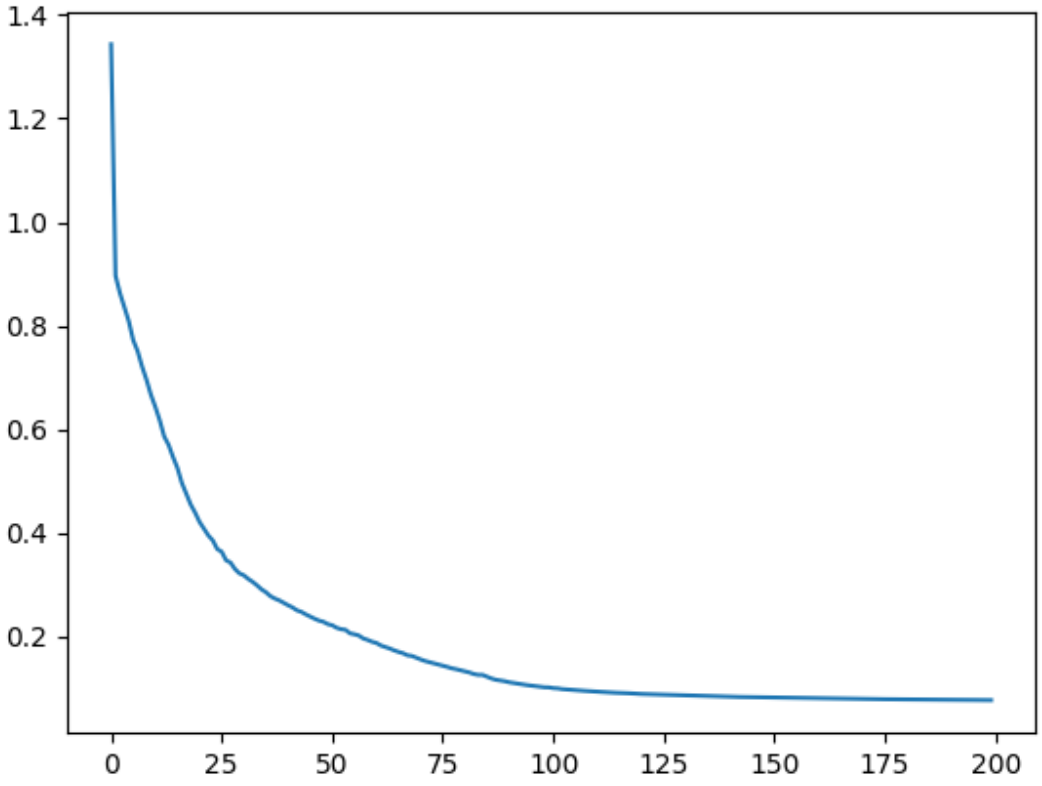
برای بخش 3 باید از شیوه حلقه ای روش گرادیان استفاده می کردیم که بسیار زمان بر بود ولی خب دقت مدل برای 100 عکس اول کمی بالاتر رفت (با توجه به اینکه تعداد تصاویر و همین طور تعداد ایپاک ها کم بود) و به 23 به طور میانگین رسید. زمان اجرای این بخش تقریبا 1.5 برابر بخش 4 بود با 0.1 تعداد ایپاک بخش 4، پس تقریبا 15 برابر کندتر از مدل پیاده سازی شده با ضرب ماتریسی عمل می کند.



Cost function for epochs = 20 , batch_size = 10

بخش 4:

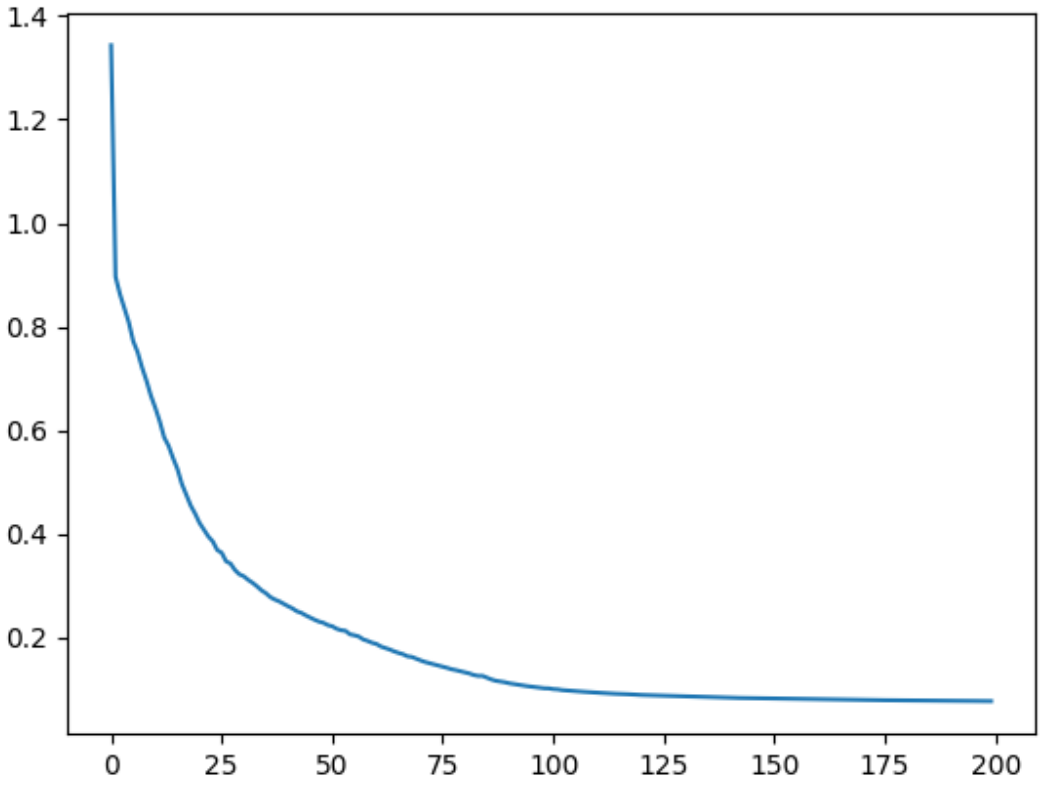
برای بخش 4 نیز صرفا باید مدل گرادیان خود را به صورت ماتریسی تبدیل میکردیم و با توجه به اینکه در این حالت سرعت بسیار بیشتر میشد میتوانستیم با 200 ایپاک تست بگیرم و به دقت 93 درصد به صورت میانگین رسیدم.



Cost function for epochs = 200 , batch_size = 10

بخش 5:

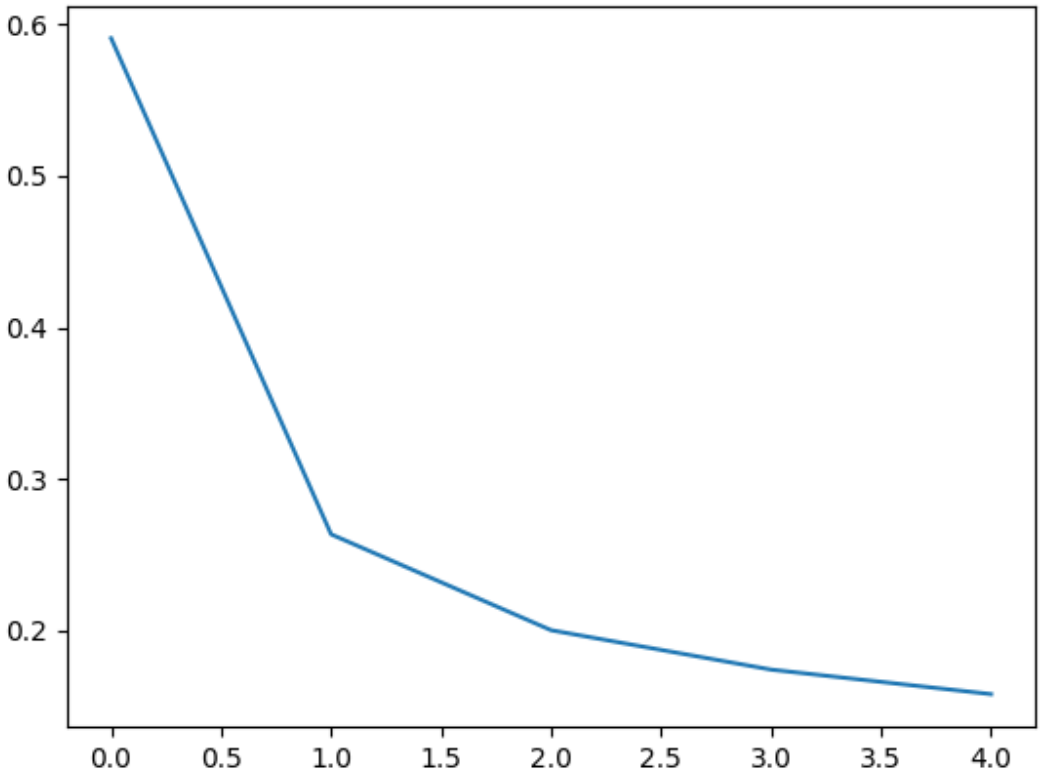
کد کاملا مشابه بخش چهار است ولی این دفعه همه داده های مجموعه آموزشی لرن میشوند و نتایج را باید بر روی داده های تست نیز به دست آوریم. با توجه به اینکه حجم اطلاعات بسیار زیاد شده تعداد ایپاک ها را کم میکنیم و به 5 میرسانیم و تعداد دسته ها را نیز به 50 برسانیم. به همین دلیل کمی دقت ما از بخش چهار پایین تر می آید و برای مجموعه تست به 89.23 و برای مجموعه داده آموزشی به 90 درصد به دست می آید.



Cost function for epochs = 5 , batch_size = 50

بخش 6 (اولین امتیازی):

با توجه به اینکه صرفا عکس های داده های تست را شیفت می دهیم دقت مدل ما به 30 درصد کاهش پیدا میکند و دلیل آن این است که این شبکه عصبی همانند ویدیو ها به این گونه نیست که ابتدا خم های شکسته را بدست آورد و سپس خم های بزرگتر و در نهایت عدد مورد نظر را تشخیص دهد. این شبکه صرفا به گونه برنامه نویسی شده ست که عکس هایی کاملا مشابه داده های آموزشی را تشخیص دهد و در صورتی که عدد ما شیفت پیدا کند با توجه به اینکه لایه اول شبکه عصبی ما به کلی تغییر میکند در نتیجه پیکسل های مختلف به وزنها و بایاس هایی متصل می شوند که به آن پیکسل خاص متعلق نبوده در نتیجه همه شبکه بهم می ریزد و دقت به شدت کاهش پیدا میکند.



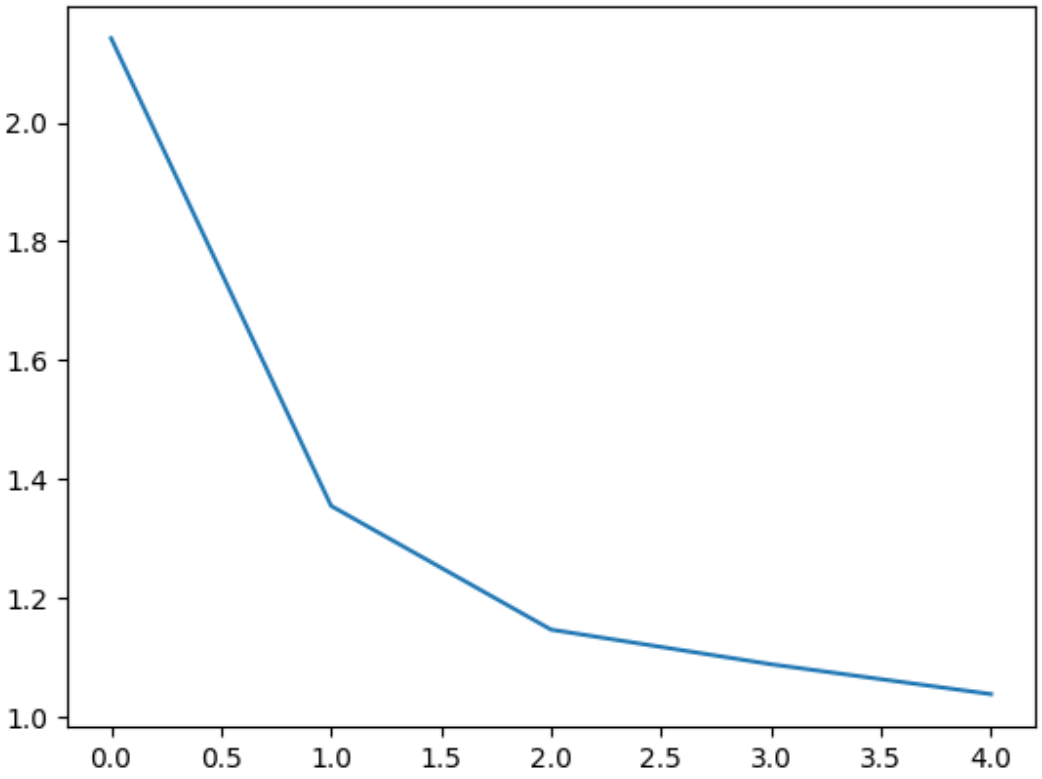
Cost function for epochs = 5 , batch_size = 50

بخش 7 (دومین امتیازی):

برای این بخش من از فعال ساز Tanh استفاده کردم و صرفا لازم بود تابع sigmoid را تغییر دهم.

دقت کاهش پیدا میکند و تقریبا به 73 درصد می رسیم و همین طور واریانس دقت هم به شدت نسبت به سیگموئید افزایش پیدا کرده و و ما بازه بزرگ تری برای دقت این مدل داریم.

مدت زمان اجرا این مدل هم تقریبا 33.33 درصد از مدل سیگموئید بهتر است و زودتر به همگرایی میرسد.



Cost function for epochs = 5 , batch_size = 50