به نام خدا

هوش مصنوعی **پروژه پنجم**

علی دارابی – شماره دانشجویی : ۸۱۰۱۰۰۲۶۴

عنوان پروژه : Convolutional Neural Networks

پیش پردازش داده

تاثیر عملیات Normalization مقدار پیکسل های عکس ها و مزایا و معایب آن :

عملیات Normalization یکی از مراحل مهم پردازش تصاویر در CNN است. Normalization ، باعث تغییر مقیاس مقادیر پیکسل های تصویر به یک محدوده مشخص میشود و معمولا بین 0 و 1 یا بین 1- و 1 است. **مزایای Normalization :** با مقیاس بندی مقادیر پیکسل به یک محدوده مشابه، الگوریتم یادگیری سریعتر همگرا میشود. مقادیر بسیار بزرگ یا بسیار کوچک پیکسل ها می تواند منجر به مشکلات عددی در محاسبات شود، نرمال کردن این مشکل را برطرف می کند. نرمال کردن باعث کاهش وابستگی بیش از حد به مقادیر خاص پیکسل ها می شود و از overfitting جلوگیری می کند. **معایب Normalization :** محاسبات اضافی برای Normalization ، هزینه محاسباتی را افزایش میدهد. در برخی موارد ممکن است اطلاعات مفیدی در مقادیر اصلی پیکسل ها وجود داشته باشد که با Normalization از بین می رود. اگر در مرحله ای به عکس اولیه نیاز داشته باشیم ، برای بازیابی تصویر اصلی نیاز به مراحل برعکس Normalizationاست.

نسبت تقسیم بندی داده های Train و Test و دلیل این انتخاب :

برای تقسیم دیتاست به دو بخش train وtest ، نسبت ۸۰:۲۰ یا ۷۰:۳۰ معمولاً توصیه می‌شود. در این پروژه ما از نسبت ۸۰:۲۰ استفاده کردیم. دلایل انتخاب این نسبت‌ها:

حجم بیشتری از داده‌ها در train set قرار میگیرد تا مدل بتواند بهتر آموزش ببیند و معمولاً حداقل ۷۰ یا ۸۰ درصد داده ها برای train لازم است.حجم کافی از داده‌ها باید در test set باقی بماند تا ارزیابی دقیقی از عملکرد مدل صورت گیرد و معمولاً حداقل ۲۰ یا ۳۰ درصد داده ها برای test نیاز است. تنظیم مناسب نسبت داده های train و test از overfitting یا underfittingمدل جلوگیری میکند. درنتیجه قاعده کلی ۸۰:۲۰ یا ۷۰:۳۰ برای اکثر مسائل نسبت خوب و قابل اطمینانی است.

طبقه بندی تصاویر با استفاده از شبکه CNN

توابع فعالسازی مختلف و تفاوت های آن ها با هم و دلیل استفاده از هریک از آنها :

ReLU: فقط مقادیر مثبت را عبور می دهد، مقادیر منفی را با تنظیم آنها بر روی صفر حذف می کند. ساده و کارآمد است.

بعد از لایه های کانولوشن و لایه های کاملا متصل استفاده می شود.

Leaky ReLU: شبیه ReLU اما به جای حذف کامل مقادیر منفی، یک شیب کوچک برای مقادیر منفی در نظر می گیرد. در اکثر موارد می تواند جایگزین ReLU شود.

ELU: تابع نمایی که اجازه خروجی های مثبت و منفی را می دهد، بر خلاف ReLU رفتاری صاف تر دارد. به عنوان جایگزینی برای ReLU در لایه های مخفی استفاده می شود ولی رایج نیست.

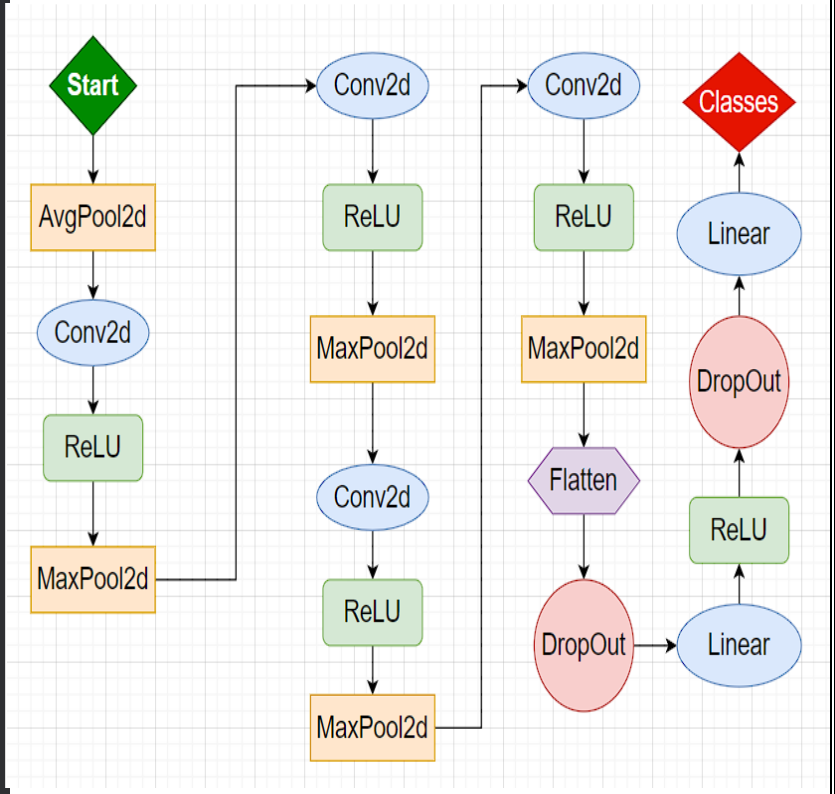
Tanh: مقادیر را بین -۱ و ۱ فشرده می کند و در مرکز صفر قرار دارد. می تواند باعث مشکلات محو شدن گرادیان شود. معمولاً در CNN های مدرن به دلیل مشکل ناپدید شدن گرادیان استفاده نمی شود. گاهی اوقات در لایه خروجی برای classification باینری استفاده می شود.

Sigmoid: مقادیر را بین ۰ و ۱ فشرده می کند. برای طبقه بندی (classification) دودویی در برخی لایه های خروجی استفاده می شود.

Softmax: خروجی ها را به توزیع احتمال برای طبقه بندی چند کلاسه تبدیل می کند. معمولاً در لایه نهایی خروجی استفاده می شود.

معماری شبکه CNN

```
self.features = nn.Sequential(
    nn.AvgPool2d(kernel_size=2),

    nn.Conv2d(3, 16, kernel_size=3, padding=1),
    #nn.BatchNorm2d(16),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(kernel_size=2, stride=2),

    nn.Conv2d(16, 32, kernel_size=3, padding=1),
    #nn.BatchNorm2d(32),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(kernel_size=2, stride=2),

    nn.Conv2d(32, 64, kernel_size=3, padding=1),
    #nn.BatchNorm2d(64),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(kernel_size=2, stride=2),

    nn.Conv2d(64, 128, kernel_size=3, padding=1),
    #nn.BatchNorm2d(128),
    nn.ReLU(inplace=True),
    nn.MaxPool2d(kernel_size=2, stride=2),
)

self.classifier = nn.Sequential(
    nn.Dropout(p=0.2),
    nn.Linear(32768, 128),
    nn.ReLU(inplace=True),

    nn.Dropout(p=0.2),
    nn.Linear(128, num_classes)
)
```



در ابتدا از یک AvgPool استفاده کردیم و kernel size آنرا برابر با ۲ گذاشتیم که اینکار باعث میشود ورودی ۵۱۲ * ۵۱۲ ما به ۲۵۶ * ۲۵۶ تبدیل شود. دلیل اینکار هم این است که باعث میشود سرعت فرایند آموزش و دقت ما افزایش یابد. (این ایده را از معماری Alex Net الهام گرفتیم.)

سپس از چهار لایه کانولوشن استفاده کردیم، با اینکار به صورت تدریجی ویژگی های پیچیده تری استخراج شود. لایه اول فیچر مپ را از سایز ۳ (چنل های RGB) به ۱۶ میبرد، در این لایه padding را برابر ۱ گذاشتیم تا اطلاعات حاشیه ای از بین نرود کرنل سایز را برابر ۳ گذاشتیم به دلیل اینکه مقدار استاندارد و مناسبی برای شبکه های تشخیص تصویر است و وقتی با مقدار های دیگر تست کردیم، دقت ما کاهش یافت. سپس از یک ReLU برای تابع فعالسازی استفاده کردیم و پارامتر آن را True گذاشتیم تا Tensor جدیدی ساخته نشود. دلیل استفاده از این تابع هم این است که ساده و کارآمد و چون نسبت به سایر توابع محاسبات کمتری دارد. سپس از یک MaxPool استفاده کردیم تا سایز را کاهش دهیم و فیچر های شایسته تر به لایه بعدی انتقال یابند، پارامتر های آن یعنی kernel size و stride را برابر با ۲ گذاشتیم تا باعث داون سمپل تدریجی شود. در سه لایه بعدی هم به همین ترتیب عمل کردیم و مقدار پارامتر هارا تغییر ندادیم و تنها در هر مرحله فیچر مپ را از ۱۶ به ۳۲ ، از ۳۲ به ۶۴ و در نهایت از ۶۴ به ۱۲۸ بردیم.

سپس در بخش بعدی ابتدا ویژگی‌های استخراج شده توسط لایه‌های Convolution را به یک بردار تبدیل کرده و با یک لایه خطی به ۱۲۸ نرون تبدیل می‌کنیم. این کار باعث کاهش تعداد پارامترها و همچنین اعمال ترکیب خطی وزن‌ها بر روی ویژگی‌ها می‌شود .سپس از یک لایه Dropout با نرخ ۰.۲ استفاده کردیم تا overfitting کاهش یابد. و در نهایت ویژگی‌ها را به تعداد کلاس‌ها (۴) تبدیل میکنیم.

توابع هزینه مختلف و نحوه عملکرد هر یک آنها :

خطای میانگین مربعات (MSE):

مقادیر پیش‌بینی و واقعی را مقایسه می‌کند، تفاوت‌ها را مربع می‌گیرد و میانگین می‌گیرد. کمینه کردن MSE باعث نزدیک شدن پیش‌بینی‌ها به مقادیر واقعی می‌شود. برای رگرسیون استفاده می‌شود.

(Cross-Entropy Loss):

فاصله بین احتمالات کلاس پیش‌بینی شده و برچسب های واقعی one-hot را اندازه می‌گیرد. کمینه کردن آنتروپی متقاطع باعث افزایش احتمال برای کلاس درست (true class) می‌شود. برای classification استفاده می‌شود.

(Hinge Loss):

پیش‌بینی‌هایی که از برچسب واقعی دور هستند، جریمه می‌کند. یک حاشیه برای مقداری انحراف اجازه داده می‌شود. برای classifier استفاده می‌شود.

(Kullback-Leibler Divergence):

میزان واگرایی توزیع احتمال پیش بینی شده از توزیع واقعی را اندازه گیری میکند. به حداقل رساندن واگرایی KL باعث می شود توزیع ها مشابه باشند. برای خروجی های احتمال استفاده می شود.

(Cosine Similarity Loss):

کسینوس زاویه بین پیش بینی ها و مقادیر واقعی را اندازه گیری می کند. زاویه کوچکتر (شباهت کسینوس بالاتر) نشان دهنده تطابق بهتر بین بردارها است. برای خروجی های با ابعاد بالا استفاده می شود.

دلیل انتخاب Cross-Entropy Loss برای پروژه :

Cross-Entropy انتخاب مرسومی برای CNN های طبقه بندی تصویر است و برای مسائل طبقه‌بندی با چند کلاس که بیش از دو کلاس خروجی دارند (برای مثال در اینجا انواع مختلف تومور) استفاده از این تابع پیشنهاد میشود. این تابع فاصله بین توزیع احتمال کلاس پیش‌بینی شده و توزیع واقعی را اندازه‌گیری می‌کند، بنابراین کمینه کردن آن باعث افزایش احتمالات پیش‌بینی شده برای کلاس‌های درست می‌شود. آنتروپی کمتر به معنای پیش‌بینی بهتر توزیع احتمالات است. همچنین به راحتی با روش‌های بهینه سازی مثل گرادیان نزولی تصادفی قابل کمینه‌سازی است.

نحوه کارکرد Adam Optimizer:

Adam یک الگوریتم بهینه سازی هوشمند است که به جای الگوریتم های سنتی مثل تنزل تصادفی (SGD) در شبکه‌های عصبی استفاده می‌شود. Adam برای هر پارامتر شبکه یک نرخ یادگیری جداگانه تنظیم می‌کند. برای این کار، میانگینی از مشتق‌های قبلی و میانگینی از مربع مشتق‌های قبلی را نگه می‌دارد. سپس با ترکیب این دو میانگین، نرخ یادگیری هر پارامتر را به‌روزرسانی می‌کند. مزیت Adam این است که نرخ یادگیری را برای هر پارامتر به صورت خودکار تنظیم می‌کند. پارامترهایی که تغییرات بیشتری دارند، نرخ یادگیری بالاتری می‌گیرند. در نتیجه سرعت همگرایی و دقت مدل بهتر می‌شود.

تفاوت Adam با الگوریتم SGD (Stochastic Gradient Descent):

SGD یک نرخ یادگیری ثابت برای همه پارامترهای مدل در نظر می‌گیرد در حالی که Adam برای هر پارامتر نرخ یادگیری مجزایی را تنظیم می‌کند. SGD به صورت تصادفی یک نمونه داده در هر مرحله انتخاب می‌کند در حالی که Adam از میانگین‌گیری از تمام گرادیان‌های مراحل قبل استفاده می‌کند. SGD به روزرسانی ساده‌ای بر اساس گرادیان جاری انجام می‌دهد در حالی که Adam از تخمین‌هایی از میانگین و واریانس گرادیان‌ها استفاده می‌کند. SGD ممکن است در بهینه‌سازی تابع‌های پیچیده مشکل داشته باشد در حالی که Adam به دلیل تنظیم خودکار نرخ یادگیری، کارایی بهتری دارد. Adam محاسبات بیشتری نسبت به SGD نیاز دارد ولی سرعت همگرایی بهتری دارد.

دلیل استفاده از بهینه ساز Adam :

نرخ یادگیری تطبیقی، efficient بودن از نظر محاسبات، همگرایی سریع و سادگی،بهینه ساز Adam را برای آموزش CNN های پیچیده مانند طبقه بندی تصاویر تومور مناسب می کند. یادگیری تطبیقی، مشکل non-convex loss surface (سطحی که دارای چندین نقطه بهینه محلی است.) را کاهش میدهد و از overfitting جلوگیری می کند.

آموزش مدل و تاثیر پارامتر Batch Size در فرآیند آموزش :

مقدار Batch Size که در طول آموزش استفاده می شود می تواند تاثیرات زیادی بر روند آموزش و عملکرد شبکه های عصبی داشته باشد برای مثال : Batch Size بزرگتر باعث بهبود کارایی محاسباتی و سخت افزاری در طول آموزش می شود زیرا نمونه های بیشتری را می توان به صورت موازی پردازش کرد. با این حال، ممکن است همگرایی آموزشی را کاهش دهد. Batch Size کوچکتر بروزرسانی‌های مکرر پارامتر را فراهم می‌کند، که منظم‌سازی بیشتری را فراهم می‌کند و به تعمیم بهتر مدل کمک می‌کند. اما از نظر محاسباتی کندتر است. اندازه خیلی کوچک Batch Size می تواند باعث ایجاد گرادیان های ناپایدار و مشکلات همگرایی شود. اندازه خیلی بزرگ هم می تواند باعث شود که مدل در یک local max که مقدار مناسبی ندارد، گیر کند. اندازه Batch Size بزرگتر همچنین واریانس به روزرسانی پارامترها را کاهش می دهد، که می تواند منجر به آموزش پایدارتر شود. اندازه Batch Size با learning rate و تعداد تکرارها در تعامل است. بنابراین تنظیم هر سه برای بهبود عملکرد آموزش CNN مهم است. بنابراین اندازه Batch Size مناسب، کارایی محاسباتی و توانایی تعمیم مدل را متعادل می کند. مقادیر معقول از ۳۲ تا ۲۵۶ متغیر است.

تاثیر روش های Regularization بر روی آموزش مدل :

Regularization ، انعطاف‌پذیری مدل را با کوچک کردن یا drop کردن بخش‌هایی از آن محدود می‌کند. این عمل Overfitting را کاهش می دهد و توانایی مدل را برای تعمیم به داده های جدید به جای به خاطر سپردن نویز در داده های آموزشی بهبود می بخشد.

Batch Normalization:

این روش با کم کردن میانگین batch و تقسیم بر انحراف استاندارد batch، ورودی های هر لایه شبکه را نرمالایز می کند. اینکار به مقابله با مشکل internal covariate shift کمک می کند، جایی که توزیع ورودی ها به هر لایه در طول آموزش با تغییر پارامترهای لایه های قبلی تغییر می کند. با re-centering و re-scaling ورودی ها، به هر لایه از شبکه اجازه می دهد تا مستقل از لایه های دیگر بهتر آموزش ببیند. وابستگی گرادیان ها به مقیاس پارامترها یا مقادیر اولیه آنها را کاهش می دهد که باعث افزایش سرعت آموزش می شود.

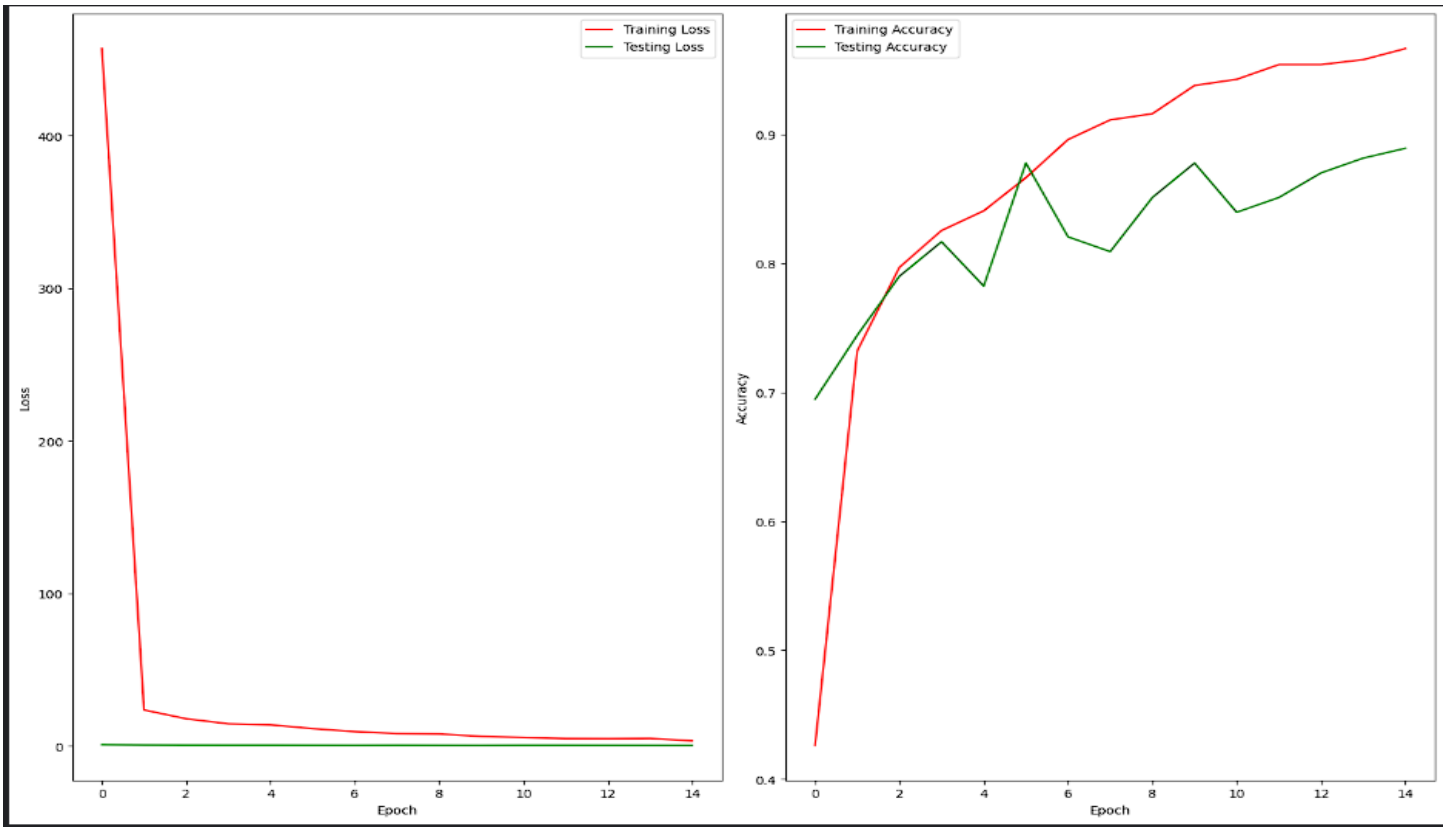
Drop Out:

یکی از روش‌های مهم Regularization در شبکه های عصبی است که بدین شکل عمل می‌کند: در این روش، در هر مرحله آموزش، تعدادی از نرون‌های یک لایه به طور تصادفی حذف (Dropout) می‌شوند. برای مثال اگر نرخ Dropout 20% تنظیم شده باشد، در هر بار آموزش ۲۰ درصد از نرون‌های آن لایه غیرفعال می‌شوند. اینکار باعث می‌شود شبکه به هیچ نرون خاصی وابسته نشود و از همه نرون‌ها به طور مستقل استفاده کند. در نتیجه Overfitting کاهش پیدا می‌کند و شبکه توانایی تعمیم بهتری پیدا می‌کند.لازم به ذکر است که در مرحله test، تمام نرون‌ها فعال هستند.

مقایسه نتایج بدون Regularization و با Regularization :

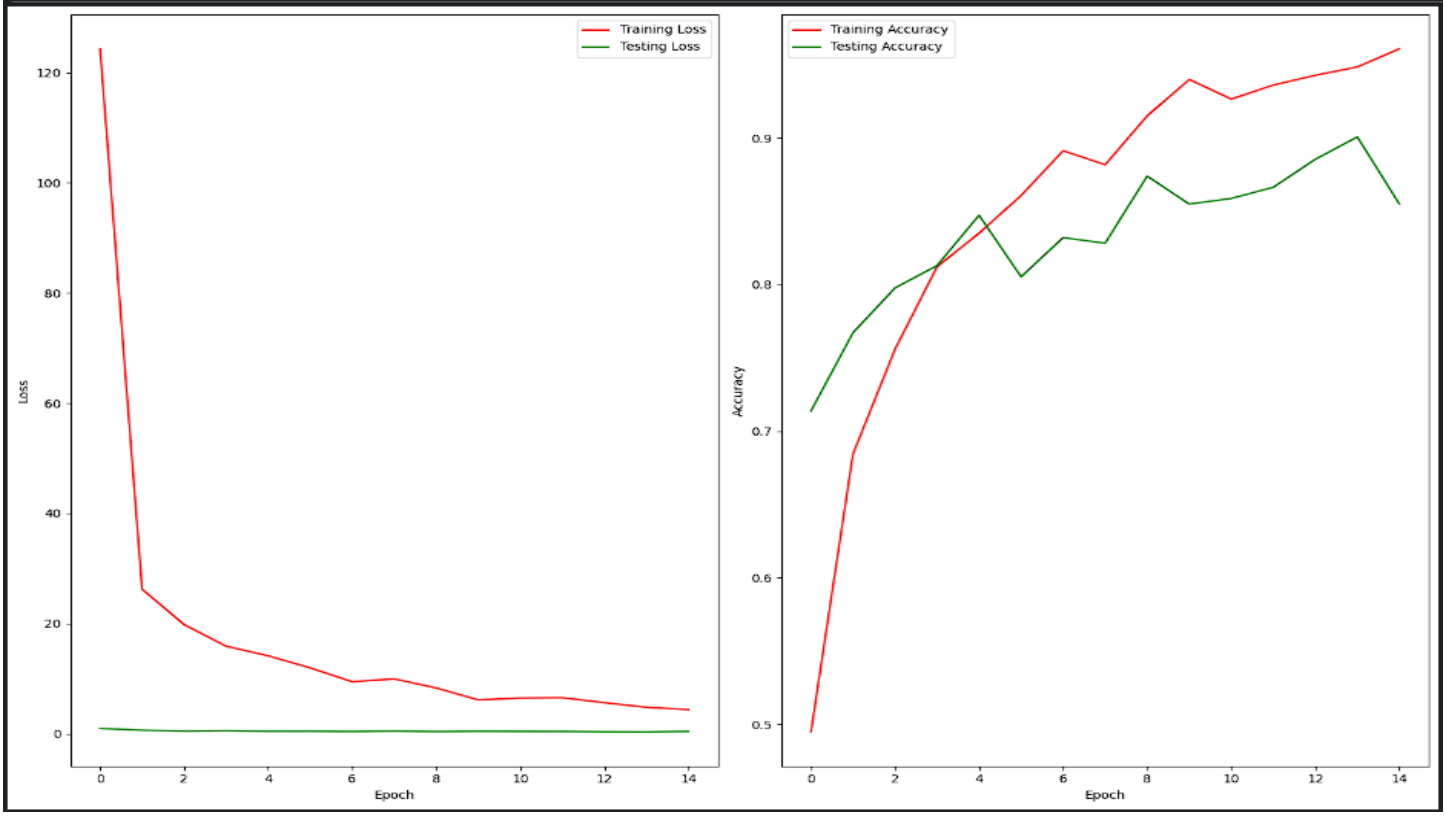
بدون اعمال Regularization :

```
100%|██████████| 33/33 [00:10<00:00,  3.12it/s]
Epoch [1/15], Train Loss: 456.93, Train Accuracy: 42.61%, Test Loss: 0.86, Test Accuracy: 69.47%
100%|██████████| 33/33 [00:10<00:00,  3.18it/s]
Epoch [2/15], Train Loss: 23.59, Train Accuracy: 73.21%, Test Loss: 0.60, Test Accuracy: 74.43%
100%|██████████| 33/33 [00:10<00:00,  3.18it/s]
Epoch [3/15], Train Loss: 17.82, Train Accuracy: 79.69%, Test Loss: 0.52, Test Accuracy: 79.01%
100%|██████████| 33/33 [00:10<00:00,  3.13it/s]
Epoch [4/15], Train Loss: 14.52, Train Accuracy: 82.55%, Test Loss: 0.49, Test Accuracy: 81.68%
100%|██████████| 33/33 [00:10<00:00,  3.10it/s]
Epoch [5/15], Train Loss: 13.83, Train Accuracy: 84.08%, Test Loss: 0.49, Test Accuracy: 78.24%
100%|██████████| 33/33 [00:10<00:00,  3.16it/s]
Epoch [6/15], Train Loss: 11.37, Train Accuracy: 86.65%, Test Loss: 0.44, Test Accuracy: 87.79%
100%|██████████| 33/33 [00:10<00:00,  3.17it/s]
Epoch [7/15], Train Loss: 9.38, Train Accuracy: 89.61%, Test Loss: 0.40, Test Accuracy: 82.06%
100%|██████████| 33/33 [00:10<00:00,  3.19it/s]
Epoch [8/15], Train Loss: 8.14, Train Accuracy: 91.13%, Test Loss: 0.46, Test Accuracy: 80.92%
100%|██████████| 33/33 [00:10<00:00,  3.11it/s]
Epoch [9/15], Train Loss: 7.86, Train Accuracy: 91.61%, Test Loss: 0.40, Test Accuracy: 85.11%
100%|██████████| 33/33 [00:10<00:00,  3.11it/s]
Epoch [10/15], Train Loss: 6.24, Train Accuracy: 93.80%, Test Loss: 0.36, Test Accuracy: 87.79%
100%|██████████| 33/33 [00:10<00:00,  3.15it/s]
Epoch [11/15], Train Loss: 5.58, Train Accuracy: 94.28%, Test Loss: 0.44, Test Accuracy: 83.97%
100%|██████████| 33/33 [00:10<00:00,  3.19it/s]
Epoch [12/15], Train Loss: 4.85, Train Accuracy: 95.42%, Test Loss: 0.44, Test Accuracy: 85.11%
100%|██████████| 33/33 [00:10<00:00,  3.21it/s]
Epoch [13/15], Train Loss: 4.77, Train Accuracy: 95.42%, Test Loss: 0.40, Test Accuracy: 87.02%
100%|██████████| 33/33 [00:10<00:00,  3.09it/s]
Epoch [14/15], Train Loss: 4.93, Train Accuracy: 95.81%, Test Loss: 0.38, Test Accuracy: 88.17%
100%|██████████| 33/33 [00:10<00:00,  3.09it/s]
Epoch [15/15], Train Loss: 3.34, Train Accuracy: 96.66%, Test Loss: 0.35, Test Accuracy: 88.93%
Best Test Accuracy: 88.93
```



با اعمال Drop Out :

```
100%|| 33/33 [00:10<00:00,  3.15it/s]
Epoch [1/15], Train Loss: 124.34, Train Accuracy: 49.48%, Test Loss: 0.96, Test Accuracy: 71.37%
100%|| 33/33 [00:10<00:00,  3.16it/s]
Epoch [2/15], Train Loss: 26.22, Train Accuracy: 68.45%, Test Loss: 0.62, Test Accuracy: 76.72%
100%|| 33/33 [00:10<00:00,  3.14it/s]
Epoch [3/15], Train Loss: 19.79, Train Accuracy: 75.60%, Test Loss: 0.47, Test Accuracy: 79.77%
100%|| 33/33 [00:10<00:00,  3.05it/s]
Epoch [4/15], Train Loss: 15.87, Train Accuracy: 81.22%, Test Loss: 0.53, Test Accuracy: 81.30%
100%|| 33/33 [00:10<00:00,  3.12it/s]
Epoch [5/15], Train Loss: 14.13, Train Accuracy: 83.51%, Test Loss: 0.44, Test Accuracy: 84.73%
100%|| 33/33 [00:10<00:00,  3.14it/s]
Epoch [6/15], Train Loss: 11.93, Train Accuracy: 86.08%, Test Loss: 0.45, Test Accuracy: 80.53%
100%|| 33/33 [00:10<00:00,  3.12it/s]
Epoch [7/15], Train Loss: 9.43, Train Accuracy: 89.13%, Test Loss: 0.39, Test Accuracy: 83.21%
100%|| 33/33 [00:10<00:00,  3.13it/s]
Epoch [8/15], Train Loss: 9.94, Train Accuracy: 88.18%, Test Loss: 0.48, Test Accuracy: 82.82%
100%|| 33/33 [00:10<00:00,  3.10it/s]
Epoch [9/15], Train Loss: 8.28, Train Accuracy: 91.52%, Test Loss: 0.37, Test Accuracy: 87.40%
100%|| 33/33 [00:10<00:00,  3.11it/s]
Epoch [10/15], Train Loss: 6.12, Train Accuracy: 93.99%, Test Loss: 0.44, Test Accuracy: 85.50%
100%|| 33/33 [00:10<00:00,  3.18it/s]
Epoch [11/15], Train Loss: 6.47, Train Accuracy: 92.66%, Test Loss: 0.41, Test Accuracy: 85.88%
100%|| 33/33 [00:10<00:00,  3.11it/s]
Epoch [12/15], Train Loss: 6.52, Train Accuracy: 93.61%, Test Loss: 0.40, Test Accuracy: 86.64%
100%|| 33/33 [00:10<00:00,  3.11it/s]
Epoch [13/15], Train Loss: 5.62, Train Accuracy: 94.28%, Test Loss: 0.32, Test Accuracy: 88.55%
100%|| 33/33 [00:10<00:00,  3.12it/s]
Epoch [14/15], Train Loss: 4.78, Train Accuracy: 94.85%, Test Loss: 0.29, Test Accuracy: 90.08%
100%|| 33/33 [00:10<00:00,  3.13it/s]
Epoch [15/15], Train Loss: 4.37, Train Accuracy: 96.09%, Test Loss: 0.41, Test Accuracy: 85.50%
Best Test Accuracy: 90.08
```



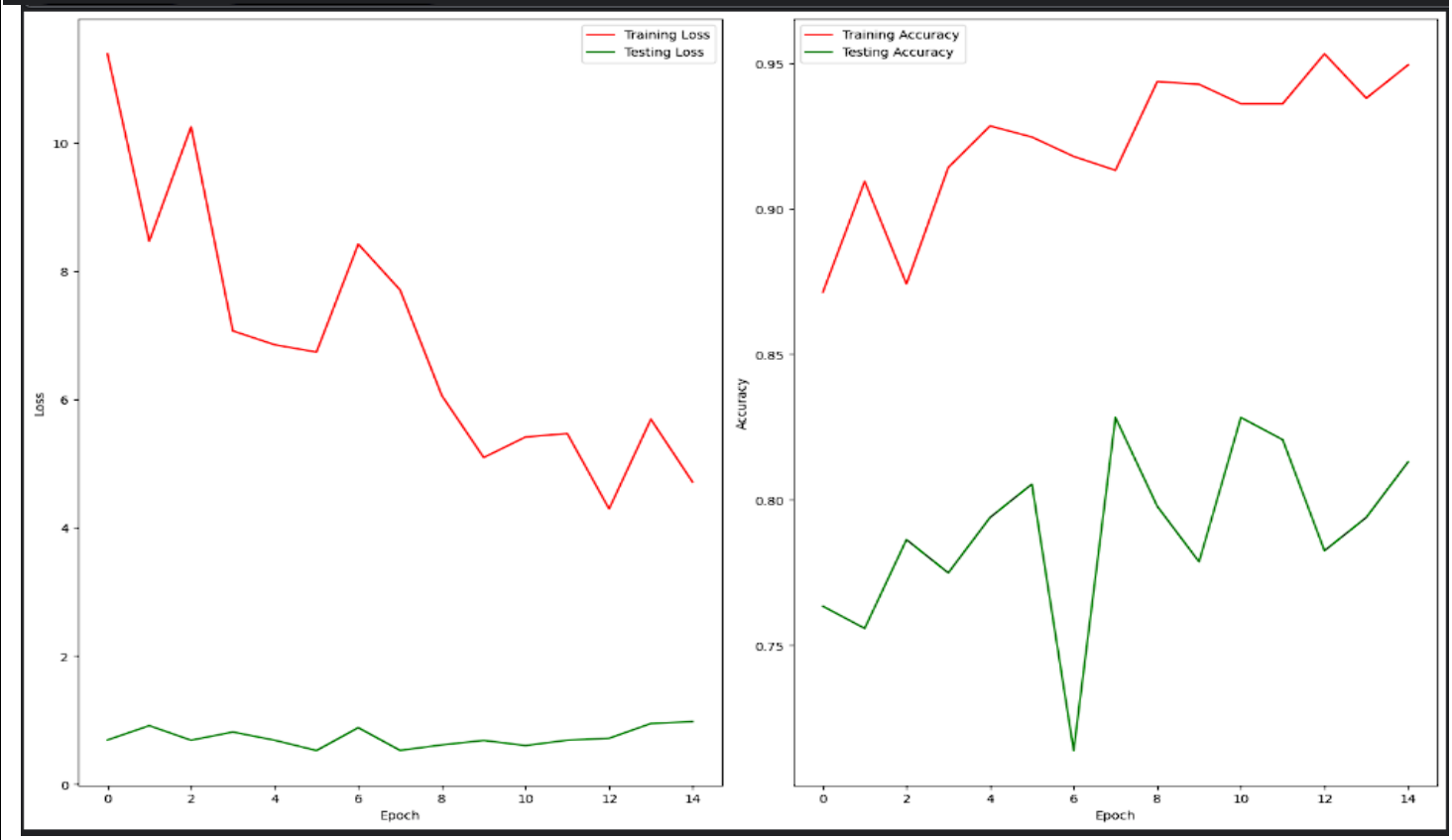
همانطور که مشاهده میکنید، زمانی که Drop Out را اعمال کردیم، تغییرات Loss و Accuracy نسبتا پایدار تر شده، و میتوان نتیجه گرفت که Drop Out باعث میشود که ما آموزش پایدار تری داشته باشیم. همچنین بالاترین دقت بدست آمده برای Test با اعمال Drop Out ، به % 90 رسید که نسبت به حالت معمولی یعنی % 88 افزایش داشت.

نرخ Drop Rate را هم ۲۰ قرار دادیم، این مقدار را بدین شکل انتخاب کردیم :

با Drop Rate، ۵۰ شروع کردیم، مقادیر ابتدایی دقت بسیار پایین بود و loss ما خیلی بالا بود، علت آن هم این بود که تعداد زیادی از نورون هارا غیرفعال کردیم. دقت آهسته آهسته در هر epoch افزایش یافته تا به یک مقدار همگرا میشد.سپس نرخ Drop کردن را در هر مرحله ۱۰ تا کاهش دادیم و مانند قبل بررسی کردیم، مقدار ۲۰ معقول ترین میزان بنظر میرسید زیرا که تعداد نورون هایی که غیر فعال میشوند، نه خیلی کم است و نه خیلی زیاد و نتایج بدست آمده بهتری داشت به همین دلیل آن را انتخاب کردیم.

با اعمال Batch Normalization :

```
100%|██████████| 33/33 [00:13<00:00,  2.53it/s]
Epoch [1/15], Train Loss: 11.39, Train Accuracy: 87.13%, Test Loss: 0.69, Test Accuracy: 76.34%
100%|██████████| 33/33 [00:12<00:00,  2.65it/s]
Epoch [2/15], Train Loss: 8.47, Train Accuracy: 90.94%, Test Loss: 0.91, Test Accuracy: 75.57%
100%|██████████| 33/33 [00:12<00:00,  2.69it/s]
Epoch [3/15], Train Loss: 10.25, Train Accuracy: 87.42%, Test Loss: 0.68, Test Accuracy: 78.63%
100%|██████████| 33/33 [00:12<00:00,  2.68it/s]
Epoch [4/15], Train Loss: 7.07, Train Accuracy: 91.42%, Test Loss: 0.81, Test Accuracy: 77.48%
100%|██████████| 33/33 [00:12<00:00,  2.69it/s]
Epoch [5/15], Train Loss: 6.85, Train Accuracy: 92.85%, Test Loss: 0.68, Test Accuracy: 79.39%
100%|██████████| 33/33 [00:12<00:00,  2.64it/s]
Epoch [6/15], Train Loss: 6.74, Train Accuracy: 92.47%, Test Loss: 0.52, Test Accuracy: 80.53%
100%|██████████| 33/33 [00:12<00:00,  2.68it/s]
Epoch [7/15], Train Loss: 8.42, Train Accuracy: 91.80%, Test Loss: 0.88, Test Accuracy: 71.37%
100%|██████████| 33/33 [00:12<00:00,  2.66it/s]
Epoch [8/15], Train Loss: 7.71, Train Accuracy: 91.33%, Test Loss: 0.52, Test Accuracy: 82.82%
100%|██████████| 33/33 [00:12<00:00,  2.68it/s]
Epoch [9/15], Train Loss: 6.06, Train Accuracy: 94.38%, Test Loss: 0.61, Test Accuracy: 79.77%
100%|██████████| 33/33 [00:12<00:00,  2.64it/s]
Epoch [10/15], Train Loss: 5.09, Train Accuracy: 94.28%, Test Loss: 0.68, Test Accuracy: 77.86%
100%|██████████| 33/33 [00:12<00:00,  2.69it/s]
Epoch [11/15], Train Loss: 5.41, Train Accuracy: 93.61%, Test Loss: 0.60, Test Accuracy: 82.82%
100%|██████████| 33/33 [00:12<00:00,  2.68it/s]
Epoch [12/15], Train Loss: 5.47, Train Accuracy: 93.61%, Test Loss: 0.68, Test Accuracy: 82.06%
100%|██████████| 33/33 [00:12<00:00,  2.67it/s]
Epoch [13/15], Train Loss: 4.29, Train Accuracy: 95.33%, Test Loss: 0.71, Test Accuracy: 78.24%
100%|██████████| 33/33 [00:12<00:00,  2.67it/s]
Epoch [14/15], Train Loss: 5.69, Train Accuracy: 93.80%, Test Loss: 0.94, Test Accuracy: 79.39%
100%|██████████| 33/33 [00:12<00:00,  2.66it/s]
Epoch [15/15], Train Loss: 4.71, Train Accuracy: 94.95%, Test Loss: 0.97, Test Accuracy: 81.30%
Best Test Accuracy: 82.82
```



همانطور که مشاهده میکنید، زمانی که از Batch Normalization استفاده میکنیم، نتایج ما به سرعت به یک مقدار همگرا میشوند. درنتیجه بعد چند epoch ابتدایی دیگر نتایج بدست آمده تغییر خاصی نمیکنند. همینطور مشاهده میشود که مقدار بدست آمده برای بهترین Test Accuracy نسبت به حالتی که از Batch Normalization استفاده نمیکنیم کاهش یافته که علت آن این است که نتایج خیلی سریع همگرا میشوند و فرصت برای بهبود ندارند.

در این مرحله ، از Batch Size = 32 استفاده کردیم. دلیل انتخاب :

وقتی از Batch Size = 16 یا Batch Size = 64 ، استفاده کردیم، مقادیر اولیه بدست آمده پایین تر بودند و چون در این حالت ، نتایج خیلی زود همگرا میشوند، نتایج نهایی هم نسبت به حالتی که Batch Size = 32 است مقادیر پایین تری بودند، برای مثال در حالتی که سایز برابر با ۱۶ بود، دقت نهایی نزدیک به ۷۶ میشد و در حالتی که سایز برابر با ۶۴ بود، دقت نهایی نزدیک به ۷۲ میشد. در نتیجه سایز ۳۲ را انتخاب کردیم.

ارزیابی و تحلیل نتایج

Confusion Matrix و معیار های خواسته شده در نوت بوک بدست آورده شده و در اینجا به بررسی آنها میپردازیم:

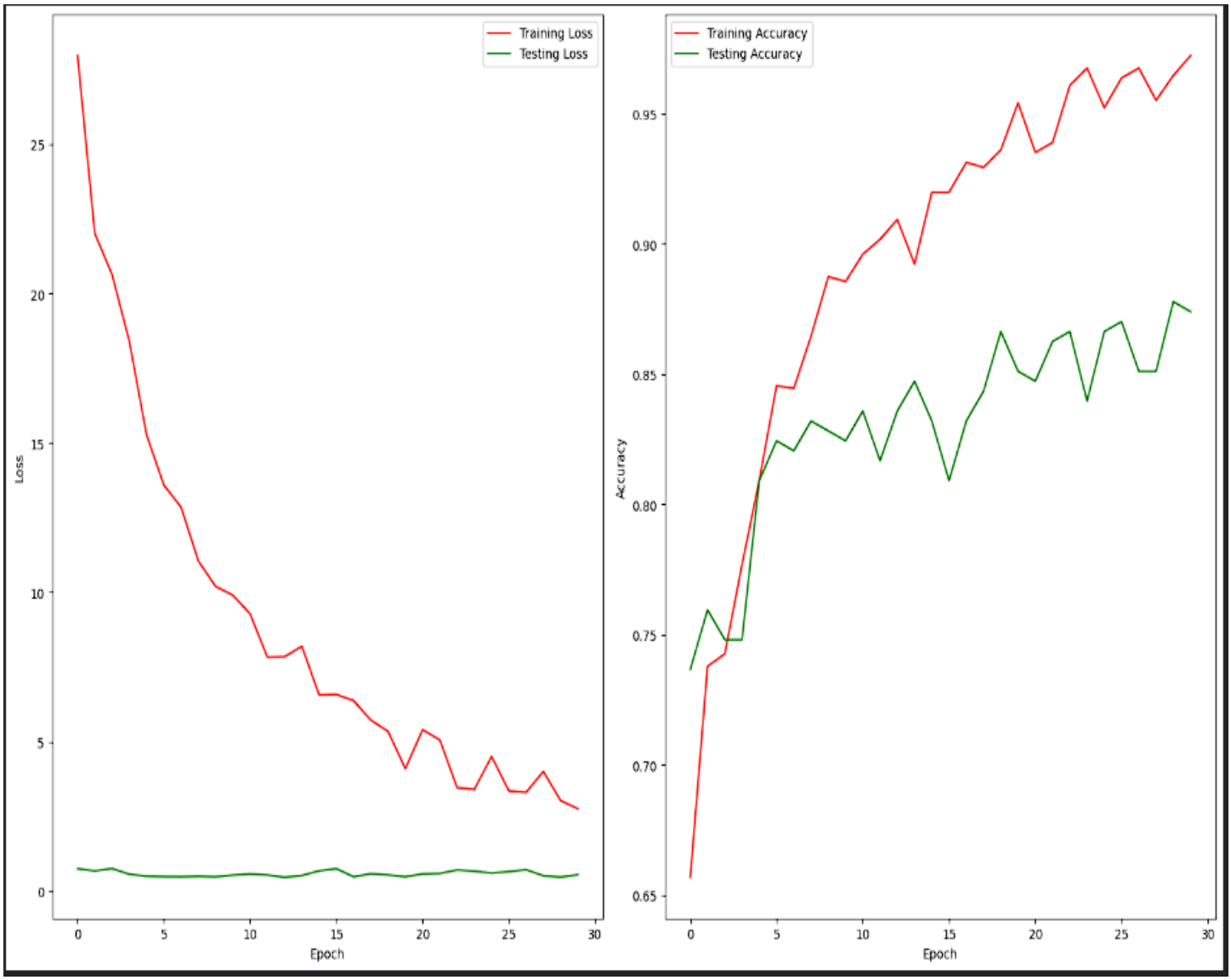
```
#Macro Average:
print("Macro Accuracy: ", (calc_accuracy(glioma_conf_mat) + calc_accuracy(meningioma_conf_mat) +
                           calc_accuracy(notumor_conf_mat) + calc_accuracy(pituitary_conf_mat))/4)
print("Macro Precision: ", (calc_precision(glioma_conf_mat) + calc_precision(meningioma_conf_mat) +
                           calc_precision(notumor_conf_mat) + calc_precision(pituitary_conf_mat))/4)
print("Macro Recall: ", (calc_recall(glioma_conf_mat) + calc_recall(meningioma_conf_mat) +
                           calc_recall(notumor_conf_mat) + calc_recall(pituitary_conf_mat))/4)
print("Macro F1 Score: ", (calc_f1_score(glioma_conf_mat) + calc_f1_score(meningioma_conf_mat) +
                           calc_f1_score(notumor_conf_mat) + calc_f1_score(pituitary_conf_mat))/4)
```

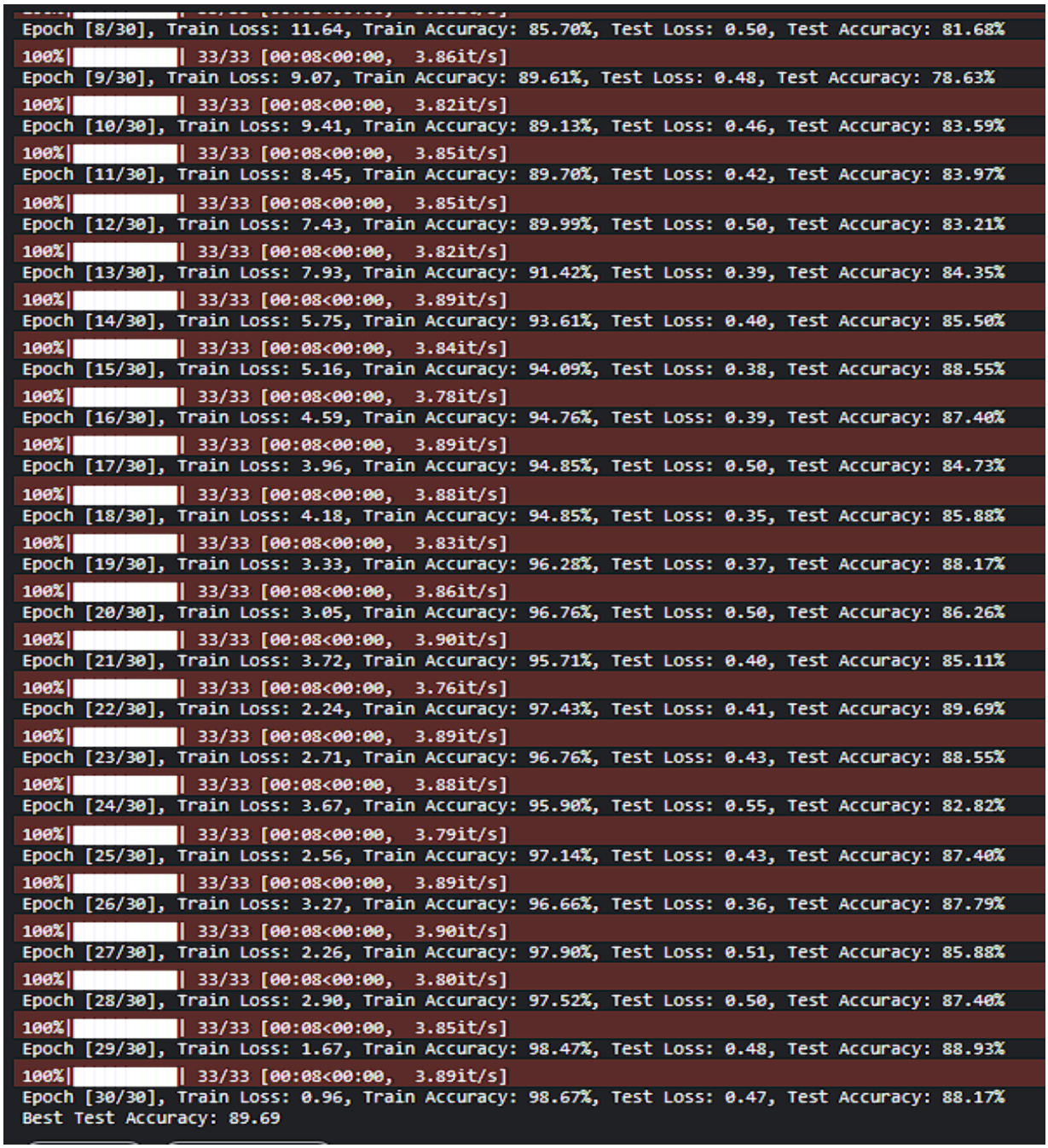
```
Macro Accuracy:  0.9484732824427481
Macro Precision:  0.895641171263434
Macro Recall:  0.8936526006881198
Macro F1 Score:  0.8942372021917477
```

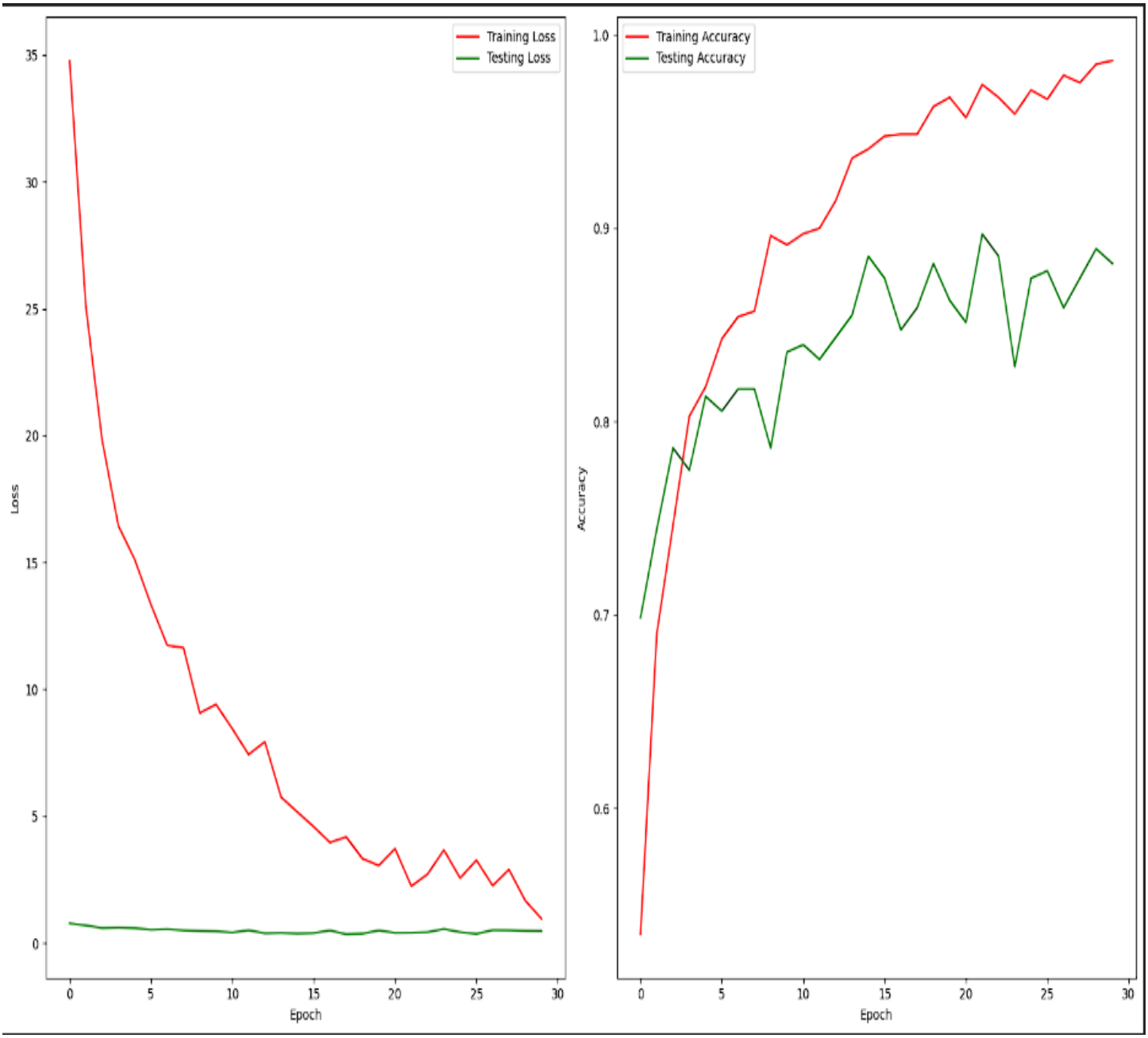
همانطور که مشاهده میکنید، نتایج بدست آمده مقادیر بالایی هستند، در نتیجه مدل ما به خوبی توانسته است داده های test را طبقه بندی کند.

نمونه ای از فرایند Train شدن مدل :

```
100%|██████████| 33/33 [00:08<00:00,  3.82it/s]
Epoch [8/30], Train Loss: 11.05, Train Accuracy: 86.46%, Test Loss: 0.50, Test Accuracy: 83.21%
100%|██████████| 33/33 [00:08<00:00,  3.75it/s]
Epoch [9/30], Train Loss: 10.20, Train Accuracy: 88.75%, Test Loss: 0.48, Test Accuracy: 82.82%
100%|██████████| 33/33 [00:08<00:00,  3.87it/s]
Epoch [10/30], Train Loss: 9.91, Train Accuracy: 88.56%, Test Loss: 0.54, Test Accuracy: 82.44%
100%|██████████| 33/33 [00:08<00:00,  3.86it/s]
Epoch [11/30], Train Loss: 9.28, Train Accuracy: 89.61%, Test Loss: 0.58, Test Accuracy: 83.59%
100%|██████████| 33/33 [00:08<00:00,  3.80it/s]
Epoch [12/30], Train Loss: 7.84, Train Accuracy: 90.18%, Test Loss: 0.54, Test Accuracy: 81.68%
100%|██████████| 33/33 [00:08<00:00,  3.88it/s]
Epoch [13/30], Train Loss: 7.85, Train Accuracy: 90.94%, Test Loss: 0.47, Test Accuracy: 83.59%
100%|██████████| 33/33 [00:08<00:00,  3.83it/s]
Epoch [14/30], Train Loss: 8.20, Train Accuracy: 89.23%, Test Loss: 0.53, Test Accuracy: 84.73%
100%|██████████| 33/33 [00:08<00:00,  3.78it/s]
Epoch [15/30], Train Loss: 6.57, Train Accuracy: 91.99%, Test Loss: 0.68, Test Accuracy: 83.21%
100%|██████████| 33/33 [00:08<00:00,  3.83it/s]
Epoch [16/30], Train Loss: 6.59, Train Accuracy: 91.99%, Test Loss: 0.76, Test Accuracy: 80.92%
100%|██████████| 33/33 [00:08<00:00,  3.86it/s]
Epoch [17/30], Train Loss: 6.38, Train Accuracy: 93.14%, Test Loss: 0.49, Test Accuracy: 83.21%
100%|██████████| 33/33 [00:08<00:00,  3.81it/s]
Epoch [18/30], Train Loss: 5.73, Train Accuracy: 92.95%, Test Loss: 0.58, Test Accuracy: 84.35%
100%|██████████| 33/33 [00:08<00:00,  3.84it/s]
Epoch [19/30], Train Loss: 5.35, Train Accuracy: 93.61%, Test Loss: 0.55, Test Accuracy: 86.64%
100%|██████████| 33/33 [00:08<00:00,  3.86it/s]
Epoch [20/30], Train Loss: 4.10, Train Accuracy: 95.42%, Test Loss: 0.49, Test Accuracy: 85.11%
100%|██████████| 33/33 [00:08<00:00,  3.79it/s]
Epoch [21/30], Train Loss: 5.41, Train Accuracy: 93.52%, Test Loss: 0.58, Test Accuracy: 84.73%
100%|██████████| 33/33 [00:08<00:00,  3.83it/s]
Epoch [22/30], Train Loss: 5.06, Train Accuracy: 93.90%, Test Loss: 0.60, Test Accuracy: 86.26%
100%|██████████| 33/33 [00:08<00:00,  3.88it/s]
Epoch [23/30], Train Loss: 3.47, Train Accuracy: 96.09%, Test Loss: 0.71, Test Accuracy: 86.64%
100%|██████████| 33/33 [00:08<00:00,  3.84it/s]
Epoch [24/30], Train Loss: 3.41, Train Accuracy: 96.76%, Test Loss: 0.67, Test Accuracy: 83.97%
100%|██████████| 33/33 [00:08<00:00,  3.83it/s]
Epoch [25/30], Train Loss: 4.51, Train Accuracy: 95.23%, Test Loss: 0.61, Test Accuracy: 86.64%
100%|██████████| 33/33 [00:08<00:00,  3.86it/s]
Epoch [26/30], Train Loss: 3.36, Train Accuracy: 96.38%, Test Loss: 0.66, Test Accuracy: 87.02%
100%|██████████| 33/33 [00:08<00:00,  3.78it/s]
Epoch [27/30], Train Loss: 3.31, Train Accuracy: 96.76%, Test Loss: 0.72, Test Accuracy: 85.11%
100%|██████████| 33/33 [00:08<00:00,  3.87it/s]
Epoch [28/30], Train Loss: 4.01, Train Accuracy: 95.52%, Test Loss: 0.52, Test Accuracy: 85.11%
100%|██████████| 33/33 [00:08<00:00,  3.84it/s]
Epoch [29/30], Train Loss: 3.03, Train Accuracy: 96.47%, Test Loss: 0.47, Test Accuracy: 87.79%
100%|██████████| 33/33 [00:08<00:00,  3.72it/s]
Epoch [30/30], Train Loss: 2.76, Train Accuracy: 97.24%, Test Loss: 0.55, Test Accuracy: 87.40%
Best Test Accuracy: 87.79
```


Training Loss
Testing Loss
Loss
Epoch
Training Accuracy
Testing Accuracy
Accuracy
Epoch

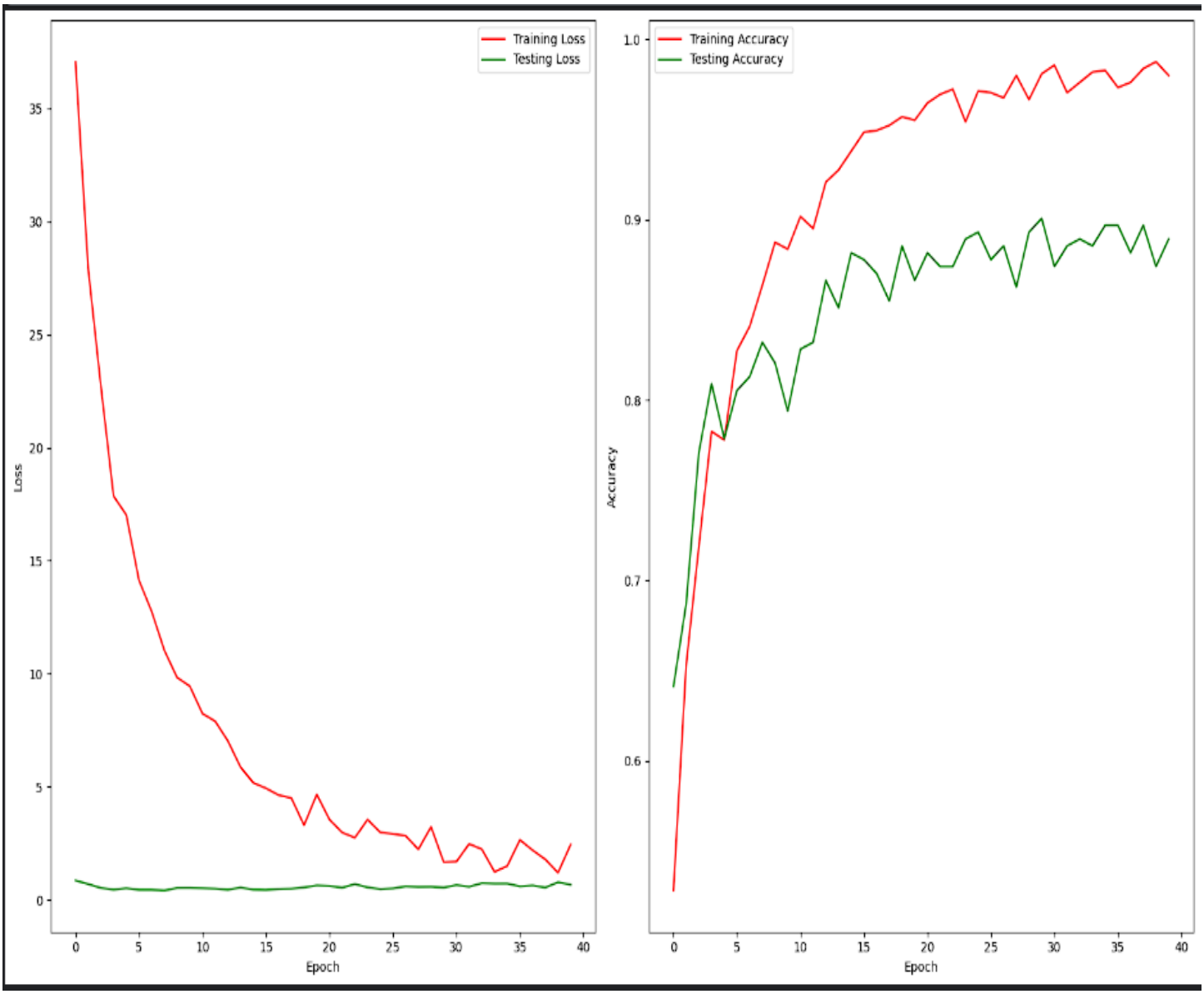

Epoch [8/30], Train Loss: 11.64, Train Accuracy: 85.70%, Test Loss: 0.50, Test Accuracy: 81.68%
100%| | 33/33 [00:08<00:00, 3.86it/s]
Epoch [9/30], Train Loss: 9.07, Train Accuracy: 89.61%, Test Loss: 0.48, Test Accuracy: 78.63%
100%| | 33/33 [00:08<00:00, 3.82it/s]
Epoch [10/30], Train Loss: 9.41, Train Accuracy: 89.13%, Test Loss: 0.46, Test Accuracy: 83.59%
100%| | 33/33 [00:08<00:00, 3.85it/s]
Epoch [11/30], Train Loss: 8.45, Train Accuracy: 89.70%, Test Loss: 0.42, Test Accuracy: 83.97%
100%| | 33/33 [00:08<00:00, 3.85it/s]
Epoch [12/30], Train Loss: 7.43, Train Accuracy: 89.99%, Test Loss: 0.50, Test Accuracy: 83.21%
100%| | 33/33 [00:08<00:00, 3.82it/s]
Epoch [13/30], Train Loss: 7.93, Train Accuracy: 91.42%, Test Loss: 0.39, Test Accuracy: 84.35%
100%| | 33/33 [00:08<00:00, 3.89it/s]
Epoch [14/30], Train Loss: 5.75, Train Accuracy: 93.61%, Test Loss: 0.40, Test Accuracy: 85.50%
100%| | 33/33 [00:08<00:00, 3.84it/s]
Epoch [15/30], Train Loss: 5.16, Train Accuracy: 94.09%, Test Loss: 0.38, Test Accuracy: 88.55%
100%| | 33/33 [00:08<00:00, 3.78it/s]
Epoch [16/30], Train Loss: 4.59, Train Accuracy: 94.76%, Test Loss: 0.39, Test Accuracy: 87.40%
100%| | 33/33 [00:08<00:00, 3.89it/s]
Epoch [17/30], Train Loss: 3.96, Train Accuracy: 94.85%, Test Loss: 0.50, Test Accuracy: 84.73%
100%| | 33/33 [00:08<00:00, 3.88it/s]
Epoch [18/30], Train Loss: 4.18, Train Accuracy: 94.85%, Test Loss: 0.35, Test Accuracy: 85.88%
100%| | 33/33 [00:08<00:00, 3.83it/s]
Epoch [19/30], Train Loss: 3.33, Train Accuracy: 96.28%, Test Loss: 0.37, Test Accuracy: 88.17%
100%| | 33/33 [00:08<00:00, 3.86it/s]
Epoch [20/30], Train Loss: 3.05, Train Accuracy: 96.76%, Test Loss: 0.50, Test Accuracy: 86.26%
100%| | 33/33 [00:08<00:00, 3.90it/s]
Epoch [21/30], Train Loss: 3.72, Train Accuracy: 95.71%, Test Loss: 0.40, Test Accuracy: 85.11%
100%| | 33/33 [00:08<00:00, 3.76it/s]
Epoch [22/30], Train Loss: 2.24, Train Accuracy: 97.43%, Test Loss: 0.41, Test Accuracy: 89.69%
100%| | 33/33 [00:08<00:00, 3.89it/s]
Epoch [23/30], Train Loss: 2.71, Train Accuracy: 96.76%, Test Loss: 0.43, Test Accuracy: 88.55%
100%| | 33/33 [00:08<00:00, 3.88it/s]
Epoch [24/30], Train Loss: 3.67, Train Accuracy: 95.90%, Test Loss: 0.55, Test Accuracy: 82.82%
100%| | 33/33 [00:08<00:00, 3.79it/s]
Epoch [25/30], Train Loss: 2.56, Train Accuracy: 97.14%, Test Loss: 0.43, Test Accuracy: 87.40%
100%| | 33/33 [00:08<00:00, 3.89it/s]
Epoch [26/30], Train Loss: 3.27, Train Accuracy: 96.66%, Test Loss: 0.36, Test Accuracy: 87.79%
100%| | 33/33 [00:08<00:00, 3.90it/s]
Epoch [27/30], Train Loss: 2.26, Train Accuracy: 97.90%, Test Loss: 0.51, Test Accuracy: 85.88%
100%| | 33/33 [00:08<00:00, 3.80it/s]
Epoch [28/30], Train Loss: 2.90, Train Accuracy: 97.52%, Test Loss: 0.50, Test Accuracy: 87.40%
100%| | 33/33 [00:08<00:00, 3.85it/s]
Epoch [29/30], Train Loss: 1.67, Train Accuracy: 98.47%, Test Loss: 0.48, Test Accuracy: 88.93%
100%| | 33/33 [00:08<00:00, 3.89it/s]
Epoch [30/30], Train Loss: 0.96, Train Accuracy: 98.67%, Test Loss: 0.47, Test Accuracy: 88.17%
Best Test Accuracy: 89.69



همانطور که مشاهده میکنید با افزایش تعداد Epoch ها، مقدار loss داده های Train روندی نزولی دارد و در Epoch های آخر به صفر میل میکند. مقدار loss داده های Test تقریبا ثابت و بسیار نزدیک به صفر است.

دقت داده های Test و Train با افزایش تعداد Epoch ها روندی صعودی داشته و هر چقدر به سمت Epoch های انتهایی میرویم ، به یک مقدار نهایی همگرا میشوند. (دقت داده های Train به سمت 100 میل میکند و دقت داده های Test در اینجا به 90 همگرا میشوند).

```
100%|          | 33/33 [00:08<00:00,  3.77it/s]
Epoch [22/40], Train Loss: 2.98, Train Accuracy: 96.95%, Test Loss: 0.54, Test Accuracy: 87.40%
100%|          | 33/33 [00:08<00:00,  3.70it/s]
Epoch [23/40], Train Loss: 2.75, Train Accuracy: 97.24%, Test Loss: 0.70, Test Accuracy: 87.40%
100%|          | 33/33 [00:08<00:00,  3.81it/s]
Epoch [24/40], Train Loss: 3.55, Train Accuracy: 95.42%, Test Loss: 0.56, Test Accuracy: 88.93%
100%|          | 33/33 [00:08<00:00,  3.78it/s]
Epoch [25/40], Train Loss: 2.99, Train Accuracy: 97.14%, Test Loss: 0.48, Test Accuracy: 89.31%
100%|          | 33/33 [00:08<00:00,  3.74it/s]
Epoch [26/40], Train Loss: 2.92, Train Accuracy: 97.04%, Test Loss: 0.51, Test Accuracy: 87.79%
100%|          | 33/33 [00:08<00:00,  3.84it/s]
Epoch [27/40], Train Loss: 2.84, Train Accuracy: 96.76%, Test Loss: 0.60, Test Accuracy: 88.55%
100%|          | 33/33 [00:08<00:00,  3.82it/s]
Epoch [28/40], Train Loss: 2.24, Train Accuracy: 98.00%, Test Loss: 0.58, Test Accuracy: 86.26%
100%|          | 33/33 [00:08<00:00,  3.72it/s]
Epoch [29/40], Train Loss: 3.23, Train Accuracy: 96.66%, Test Loss: 0.58, Test Accuracy: 89.31%
100%|          | 33/33 [00:08<00:00,  3.79it/s]
Epoch [30/40], Train Loss: 1.67, Train Accuracy: 98.09%, Test Loss: 0.55, Test Accuracy: 90.08%
100%|          | 33/33 [00:08<00:00,  3.82it/s]
Epoch [31/40], Train Loss: 1.69, Train Accuracy: 98.57%, Test Loss: 0.66, Test Accuracy: 87.40%
100%|          | 33/33 [00:08<00:00,  3.69it/s]
Epoch [32/40], Train Loss: 2.48, Train Accuracy: 97.04%, Test Loss: 0.58, Test Accuracy: 88.55%
100%|          | 33/33 [00:08<00:00,  3.84it/s]
Epoch [33/40], Train Loss: 2.24, Train Accuracy: 97.62%, Test Loss: 0.74, Test Accuracy: 88.93%
100%|          | 33/33 [00:08<00:00,  3.82it/s]
Epoch [34/40], Train Loss: 1.24, Train Accuracy: 98.19%, Test Loss: 0.72, Test Accuracy: 88.55%
100%|          | 33/33 [00:08<00:00,  3.76it/s]
Epoch [35/40], Train Loss: 1.50, Train Accuracy: 98.28%, Test Loss: 0.72, Test Accuracy: 89.69%
100%|          | 33/33 [00:08<00:00,  3.79it/s]
Epoch [36/40], Train Loss: 2.66, Train Accuracy: 97.33%, Test Loss: 0.60, Test Accuracy: 89.69%
100%|          | 33/33 [00:08<00:00,  3.84it/s]
Epoch [37/40], Train Loss: 2.20, Train Accuracy: 97.62%, Test Loss: 0.64, Test Accuracy: 88.17%
100%|          | 33/33 [00:08<00:00,  3.74it/s]
Epoch [38/40], Train Loss: 1.79, Train Accuracy: 98.38%, Test Loss: 0.55, Test Accuracy: 89.69%
100%|          | 33/33 [00:08<00:00,  3.78it/s]
Epoch [39/40], Train Loss: 1.20, Train Accuracy: 98.76%, Test Loss: 0.78, Test Accuracy: 87.40%
100%|          | 33/33 [00:08<00:00,  3.80it/s]
Epoch [40/40], Train Loss: 2.45, Train Accuracy: 98.00%, Test Loss: 0.67, Test Accuracy: 88.93%
Best Test Accuracy: 90.08
```


Training Loss
Testing Loss
Loss
Epoch
Training Accuracy
Testing Accuracy
Accuracy
Epoch