تمرین شماره دو

شبکه عصبی پیچشی(CNN)

سید سروش مجد

۴۰۰۴۴۳۱۸۱

درس یادگیری عمیق | بهار ۱۴۰۱

استاد درس: جناب آقای دکتر حامد ملک

اردیبهشت ۱۴۰۱

بخش اول؛ سوالات:

۱. مزایای استفاده از CNN در تسک‌های پردازش تصویر چیست؟

برتری اصلی CNN در تسک تصویر این است که اتوماتیک ویژگی‌های مهم را بدون نظارت انسان تشخیص می‌دهد. برای مثال، با توجه به تصاویر بسیاری از گربه‌ها و سگ‌ها، ویژگی‌های متمایز هر کلاس را به تنهایی یاد می‌گیرد. سیستم یاد می‌گیرد که استخراج ویژگی را انجام دهد و مفهوم اصلی CNN این است که از کانولوشن تصویر و فیلترها برای تولید ویژگی‌های Invariant استفاده می‌کند که به لایه بعدی منتقل می‌شوند. ویژگی‌های لایه بعدی با فیلترهای مختلف کانوالو می‌شوند تا ویژگی‌های Invariant بیشتری تولید کنند و این روند تا زمانی ادامه می‌یابد که ویژگی نهایی به دست آید. همچنین تصاویر ابعاد بالایی دارند (زیرا هر پیکسل به عنوان یک ویژگی در نظر گرفته می شود) و CNN ها در کاهش تعداد پارامترها بدون از دست دادن کیفیت مدل بسیار موثر می‌باشند. CNNها پشتیبانی دو بعدی را نیز فراهم می‌کنند. مدل‌های داده می‌توانند موقعیت‌ها و مقیاس‌ها را بیاموزند و به آن‌ها اجازه می‌دهند هنگام پردازش تصاویر به درستی عمل کنند. این شبکه از نظر محاسباتی نیز کارآمد است.

۲. آنچه در فرآیند Batch Normalization رخ می‌دهد را به صورت مختصر توضیح دهید و مزایای استفاده از آن در شبکه‌های عصبی عمیق را نیز بیان نمایید.

می‌دانیم که Normalization می‌تواند سرعت یادگیری را زیاد کند. Batch Normalization همین کار را برای لایه‌های مخفی انجام می‌دهد و برای Mini Batchها انجام می‌دهد و خروجی‌های لایه‌ها را به میانگین ۰ و واریانس ۱ مقیاس می‌کند. از مزایای آن می‌توان به افزایش سرعت آموزش و نرخ یادگیری، وزن‌دهی اولیه آسان‌تر، Regularize کردن مدل، جلوگیری از Overfitting به دلیل اضافه کردن نویز و کسب نتایج بهتر اشاره کرد. نحوه نرمالایز کردن:

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

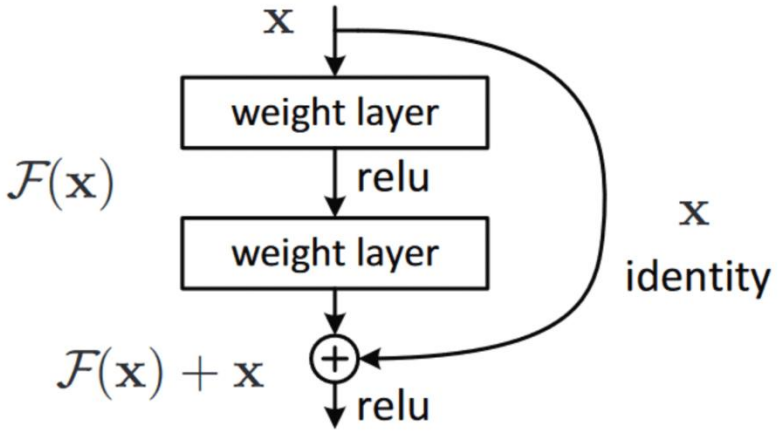
$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

۳. توضیح دهید چرا Residual Learning از محو شدگی گرادیان (Vanishing Gradient) جلوگیری می‌کند؟

یکی از جدیدترین و موثرترین راه‌ها برای حل مشکل محو شدگی گرادیان، شبکه‌های عصبی Residual است. در شبکه های عصبی Residual اتصالات پرش یا اتصالات Residual بخشی از معماری شبکه هستند. این اتصالات به گرادیان اجازه می‌دهد از لایه‌ها عبور کند (خروجی لایه قبلی به خروجی لایه عمیق تر اضافه می شود). این اجازه می دهد تا اطلاعات قسمت‌های قبلی شبکه به بخش های عمیق تر شبکه منتقل شود و به حفظ انتشار سیگنال حتی در شبکه های عمیق تر کمک کند. اتصالات Residual یک جزء حیاتی و مهم است که امکان آموزش موفقیت آمیز شبکه‌های عصبی عمیق را فراهم می‌کند.



بخش دوم; پیاده‌سازی

انتخاب تعداد فیچر ها به صورت دلخواه انجام نمی‌شود و می‌تواند به صورت شهودی یا تجربی آن‌ها را ست کـرد و متناسب با کاربرد تعداد فیچرها میتوانند متفاوت باشند برای مجموعه داده‌های پیچیده تر انتظـار میـرود کـه تعداد فیچر ها افزایش پیدا کند از طرفی تعداد فیچر ها تعداد feature map هاست اگر این تعـداد کـم باشـد یعنی تعداد ویژگی هایی که از داده‌ها استخراج کرده‌ایم کم است در نتیجه شبکه نمی‌تواند رابطه معناداری بـین داده‌ها پیدا کند و اگر این تعداد بیش از اندازه زیاد باشد مفهـوم فاصـله در feature space از بـین مـی‌رود و نتیجه مطلوبی نخواهیم داشت.

برای پیاده‌سازی از Optimizer آدام استفاده کردیم. آدام بهترین نتیجه را دارد ترکیبی از الگوریتم مومنتوم و RMSprop است و می‌دانیم که در الگوریتم مومنتوم اپدیت وزن ها بر اساس گرادیان های قبلی و گرادیان فعلی انجام می‌شود که این میانگین گیری باعث میشود که نوسانات عمودی کاهش یابد و با شدت بیشتری به سمت بهینه محلی حرکت کند

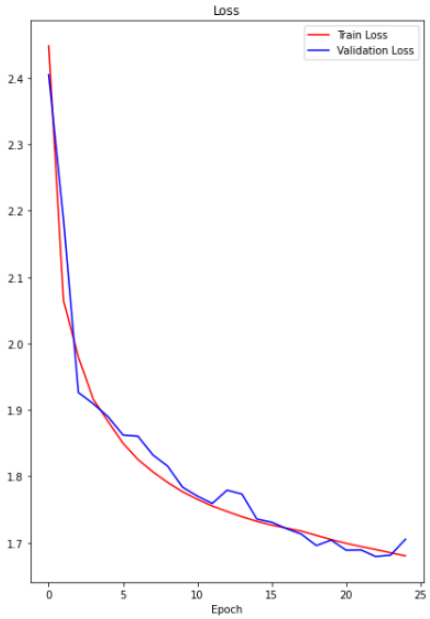
الگوریتم اداگرد (ازمعکوس ، مجموع مجزور گرادیان ها برای مقدار پارامتر یادگیر استفاده می‌کند) همیشه جواب خوبی نمی‌دهد چون کل هیستوری گرادیان ها از اول تا آنجا برای ما مهم نیست و آن چیزی که بیشترین اهمیت را دارد، گرادیان‌های اخیر هستند و جمع مربع گرادیان‌ها از ابتدای اموزش، منجر به کاهش زودرس و بیش از حد پارامتر یادگیر می‌شود در نتیجه اپدیت به کندی انجام می‌شود . پس بهتر است به جای استفاده از کل هیستوری ، وزنی را به گرادیان‌های اخیر بدهیم تا تاثیر آنها بیشتر شود که این مشکل در RMSprop حل شده است که این الگوریتم در مسایل غیر محدب هم جواب خوبی می‌دهد . الگوریتم آدام چون ترکیبی از این دو الگوریتم است پس مشکلات کمتری دارد و نتیجه بهتری را می‌دهد.

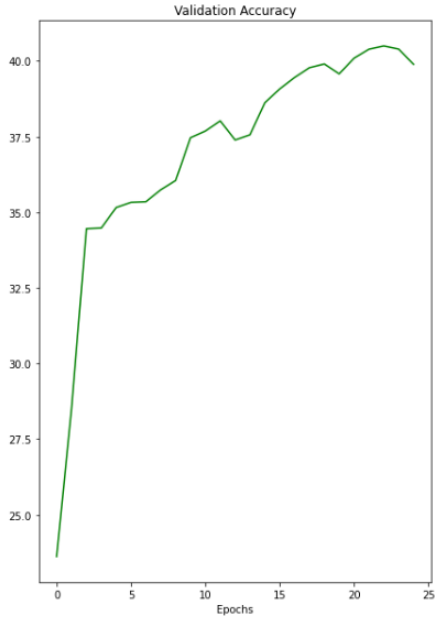
همچنین اگر لرنینگ ریت خیلی کم باشد باعث می‌شود شبکه خیلی کند همگرا باشد یا در local minimum گرفتار شود. زیاد بودن آن ممکن است باعث شود اصلا همگرا نشود و global minimum پیدا نشود. با امتحان کردن مقادیر مختلف لرنینگ ریت مناسب را به دست آوردیم تا این مشکل کم یا زیاد بودن آن به وجود نیاید. تعداد ایپاک های ما ۲۵ تا است. اگر تعداد ایپاک ها خیلی کم باشد مدل ما به خوبی اموزش نمی‌بیند و عملکرد خوبی ندارد. اگر خیلی زیاد باشد باعث می‌شود مدل خیلی اموزش ببیند. این موجب می‌شود مدل به داده های آموزشی خیلی بایاس شود و مشکل overfit به وجود آید. (دیتای آموزش را حفظ می‌کند). در این حالت نیز مانند لرنینگ ریت، تعداد ایپاک‌ها نه باید خیلی کم باشد و نه خیلی زیاد که مشکلی به وجود نیاید.

❖ Image Colorization

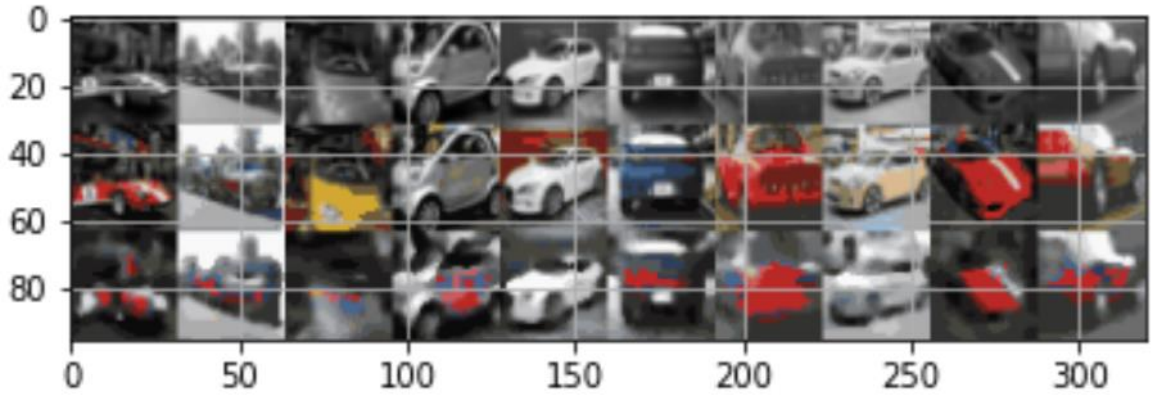
```
Epoch [25/25], Val Loss: 1.7050, Val Acc: 39.9%, Time(s): 17.10
```

دقت در این حالت با پیاده‌سازی شبکه عصبی عادی به حدود ۴۰ درصد رسید. نمودار validation loss، validation accuracy و test loss در نمودار های زیر نشان داده شده اند.





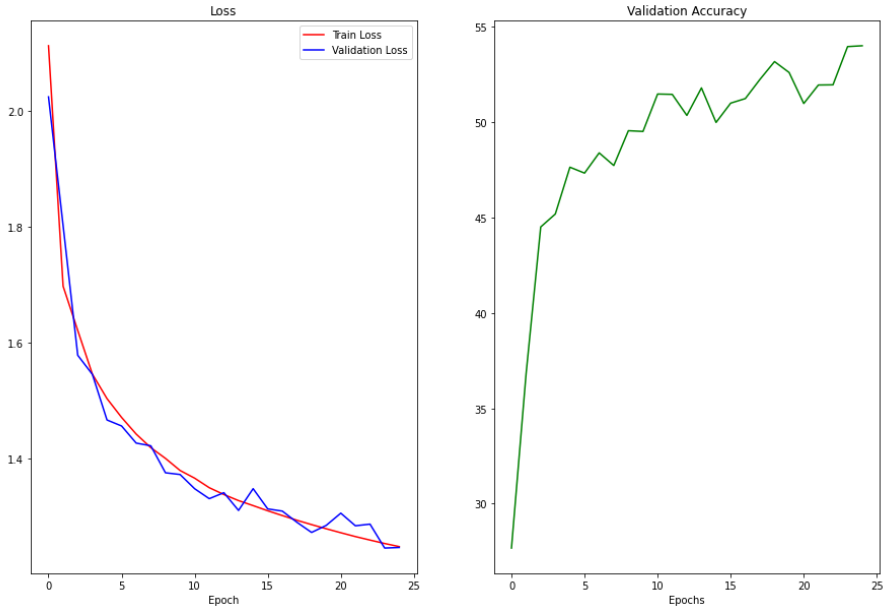
در شکل زیر ۳ حالت عکس های خام و سیاه سفید، عکس های رنگ شده موجود در دیتاست و در حالت اخر عکس هایی که ما رنگ کردیم نشان داده شده است و می‌توان آن هارا با یکدیگر مقایسه کرد. مشاهده می‌شود که دقت شبکه عصبی پیاده‌سازی شده خیلی زیاد نیست و ماشین‌ها خیلی خوب رنگ نشده‌اند.



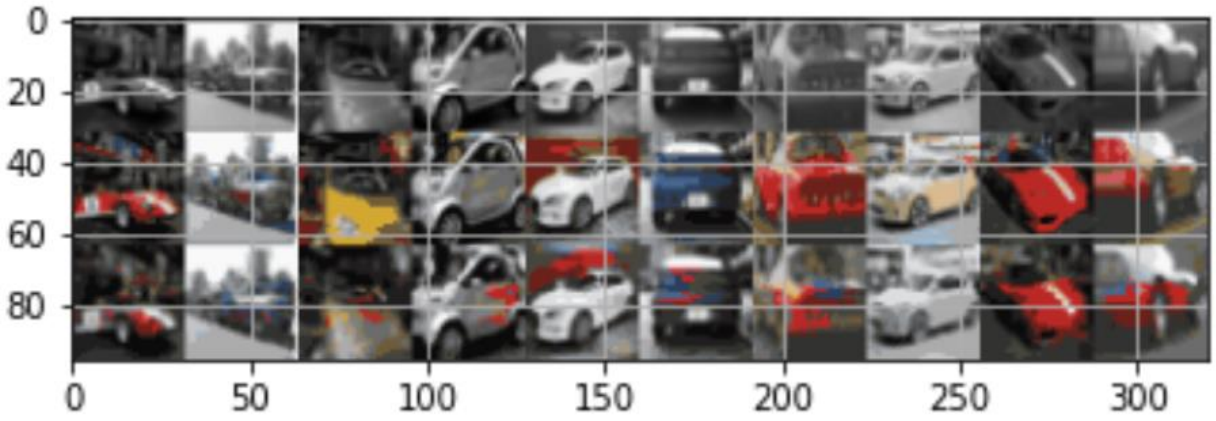
❖ Skip connection

```
Epoch [25/25], Val Loss: 1.2454, Val Acc: 54.0%, Time(s): 17.62
```

دقت در این حالت با پیاده‌سازی شبکه UNet به حدود ۵۵ درصد رسید. نمودار validation loss، validation accuracy و test loss در نمودار های زیر نشان داده شده اند.

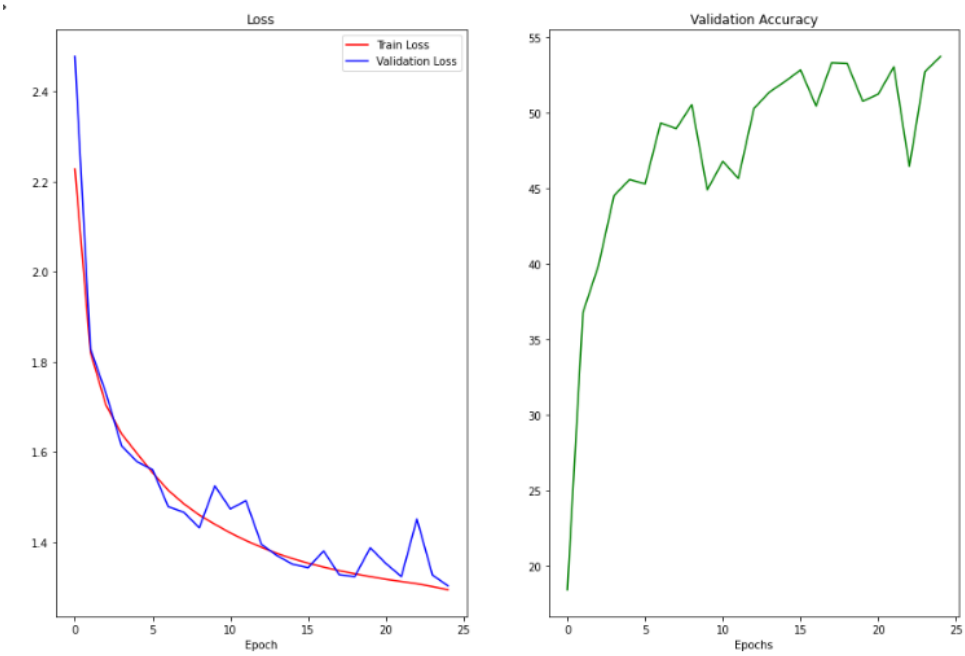


در شکل زیر ۳ حالت عکس های خام و سیاه سفید، عکس های رنگ شده موجود در دیتاست و در حالت اخر عکس هایی که ما رنگ کردیم نشان داده شده است و می‌توان آن هارا با یکدیگر مقایسه کرد. مشاهده می‌شود که دقت شبکه پیاده‌سازی شده بهتر شده‌است و ماشین‌ها خیلی بهتر رنگ شده اند. بخاطر اینکه فیچرمپ ها از لایه‌های مختلف به لایه اخر وارد میشن باعث میشود در لایه-های اخر فقط ب فیچرهای ریز بایاس نشویم و فیچرهای اولیه ک در لایهای ابتدایی استخراج شده اند را هم در نظر بگیریم. به همین خاطر نسبت به شبکه قبلی عملکرد بهتری دارد.



❖ بخش امتیازی:

```
Epoch [25/25], Val Loss: 1.3018, Val Acc: 53.7%, Time(s): 20.81
```



در شکل زیر ۳ حالت عکس‌های خام و سیاه سفید، عکس های رنگ شده موجود در دیتاست و در حالت اخر عکس هایی که ما رنگ کردیم نشان داده شده است و می‌توان آن هارا با یکدیگر مقایسه کرد. مشاهده می‌شود که دقت شبکه پیاده‌سازی شده خوب است و مانند حالت UNet بدون Residual می‌باشد. این به این دلیل است که تعداد لایه‌ها آنقدر زیاد نیست که مشکل Vanishing Gradient رخ دهد.