شبکه‌های عصبی

دانشگاه فردوسی مشهد

تمرین سه

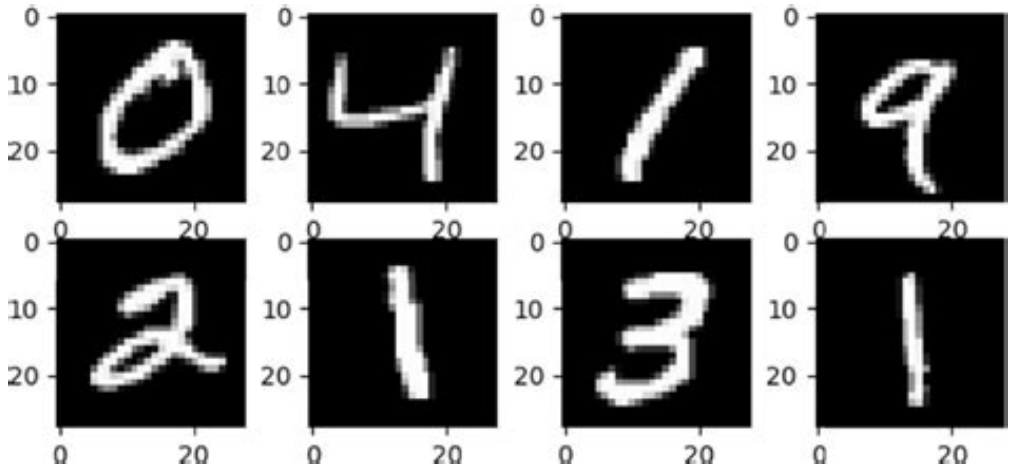
نیمسال دوم تحصیلی ۱۴۰۳-۱۴۰۲

گروه مهندسی کامپیوتر مهلت ارسال: ۱۴۰۲/۰۱/۲۰ ۲۳:۵۹

در این تمرین قصد داریم با استفاده از شبکه MLP پیاده سازی شده در تمرین اول، داده های MNIST را طبقه بندی کنیم. تا اینجا در تمرین اول و دوم با مقدمات MLP در مسائل binary classification آشنا شدید. در این تمرین هدف این است تا با استفاده از یک مسئله multi-class classification تکنیک های مقدماتی دیگری را بررسی کنید. تمرین را با استفاده از توابع استاندارد Python و NumPy و بدون استفاده از toolbox های مربوط به یادگیری عمیق پیاده سازی کنید.

دیتاست MNIST

دیتاست MNIST این دیتاست شامل ۶۰۰۰۰ عکس به عنوان داده آموزش و ۱۰۰۰۰ عکس به عنوان داده تست است و داده ها مربوط به اعداد ۰ تا ۹ است می‌شود. عکس ها به صورت grayscale هستند و ابعاد آنها ۲۸ در ۲۸ می‌باشد و مقادیر پیکسل ها از ۰ تا ۲۵۵ است.



شکل ۱: چند نمونه از داده های MNIST.

برای این تمرین، داده های MNIST با فرمت npz در اختیار شما قرار داده شده است. برای لود کردن داده ها از دستورات زیر استفاده کنید.

```
x_train = np.load('x_train.npz')['arr_0']
x_test = np.load('x_test.npz')['arr_0']
y_train = np.load('y_train.npz')['arr_0']
```

y_test = np.load('y_test.npz')['arr_0']

۱) Softmax

در مسئله binary classification در لایه آخر، یک تابع فعالساز sigmoid باعث می‌شود تا خروجی بین ۰ و ۱ و حالتی احتمالی داشته باشد. softmax این ایده را برای طبقه بندی چند کلاسه بسط میدهد. softmax یک تابع فعالساز است که معمولا در لایه آخر شبکه قرار می‌گیرد و logit ها را به یک توزیع احتمالی تبدیل می‌کند و فرمول آن به صورت زیر می‌باشد.

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_i}}$$

در ابتدا این تابع فعالساز را پیاده سازی کنید و برای لایه آخر شبکه خود از آن استفاده کنید. نتایج آموزش و تست شبکه را در حالتی که برای لایه های ابتدایی از تابع فعالساز ReLU و در لایه آخر از softmax استفاده کرده اید گزارش کنید. از آنجایی که داده های MNIST 28×28 می‌باشد، نیاز است تا داده ها را flatten کرده تا به یک بردار 784×1 تبدیل شود.

۲) پیش پردازش

یک مرحله ابتدایی قبل از آموزش شبکه، پیش پردازش اطلاعات است که تاثیرات متفاوتی بر روی یادگیری شبکه می‌گذارد. پیش پردازش می‌تواند شامل normalize کردن، standardize کردن و یا روش های دیگری شود. نرمالایز کردن شامل تغییر scale به بازه ۰ تا ۱ می‌شود و استاندارد کردن برای تغییر میانگین و انحراف معیار به ۰ و ۱ می‌باشد.

برای نرمالایز کردن ،MNIST از آنجایی که مقادیر پیکسل ها بین ۰ تا ۲۵۵ است، کافی است تا آنهارا بر ۲۵۵ تقسیم کنیم و برای استاندارد کردن می‌توان از فرمول زیر استفاده کرد:

$$x_i = \frac{x_i - \mu}{\sigma}$$

بعد از پیاده سازی روش های پیش‌پردازش گفته شده، حالت های زیر را بررسی کنید و با نتیجه سوال قبل

که بدون استفاده از هیچ روش پیش پردازشی بود مقایسه کنید و در گزارش خود نتایج را شرح دهید.

- استفاده از normalization به تنهایی
- استفاده از standardization به تنهایی
- ابتدا نرمال کردن و سپس استاندارد کردن

این موارد را با تنظیمات ثابت همانند سوال قبل (استفاده از ReLU و softmax) انجام دهید. نرخ یادگیری، تعداد لایه ها و تعداد نورون های هر لایه را نیز ثابت در نظر بگیرید و بهترین حالت ممکن را به عنوان تنظیمات دیفالت در نظر بگیرید. در تمرین های بعدی نیز برای آزمایشات خود از این تنظیمات استفاده کنید.

۳) Vanishing Gradients

شبکه را با استفاده از تابع فعالساز sigmoid و با تعداد لایه های متفاوت آموزش دهید و اندازه بردار های وزن را در هر لایه در آخر هر epoch بررسی کنید. نتایج خود را گزارش دهید.

۴) Regularization

همان‌طور که در درس خواندید، یک مشکلی که در آموزش شبکه های عصبی وجود دارد، مشکل overfitting است. برای جلوگیری از overfit شدن روش های regularization استفاده می‌شوند که در این قسمت به برخی از آنها اشاره می‌کنیم.
بعد از پیاده سازی سه روش گفته شده در زیر، عملکرد آنها را در تنظیماتی یکسان (استفاده از ReLU و softmax) با یکدیگر مقایسه کرده و در گزارش خود شرح دهید.

Batch Normalization

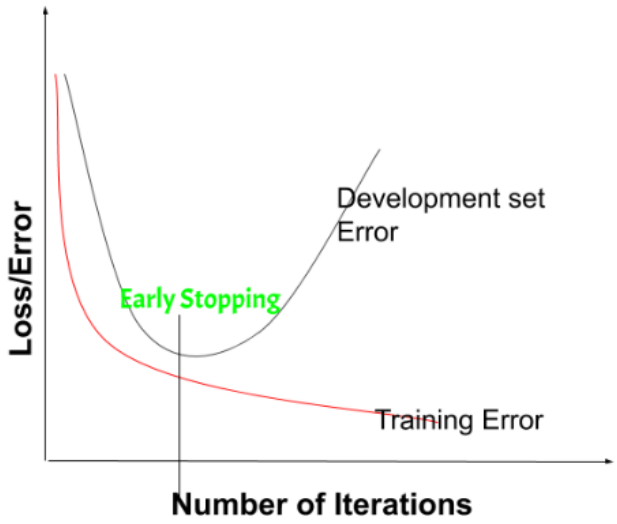
Batch normalization را می‌توان انجام دادن پیش پردازش برای هر لایه در شبکه در نظر گرفت. به این صورت که برای هر لایه ورودی ها را به توزیعی با میانگین صفر و واریانس واحد تبدیل می‌کند و باعث بهتر شدن همگرایی در یادگیری می‌شود.
برای پیاده سازی لایه batch-norm نیاز است تا forward pass و backward pass آن‌را پیاده سازی کنیم. computational graph مربوط به batch normalization را می‌توانید در این لینک مشاهده کنید و با استفاده از آن پیاده سازی را انجام دهید. توجه داشته باشید که عملکرد batch-norm در زمان آموزش و تست متفاوت است.

Dropout

یکی از راه های regularize کردن می‌باشد و شامل حذف کردن برخی از node ها در شبکه است که باعث می‌شود به صورت خاص یک node تاثیر زیادی بر روی خروجی نگذارد.
در پیاده سازی dropout باید به این توجه کرد که در زمان آموزش شبکه فقط باید حذف نورون ها اعمال شود و در زمان تست، تمامی نورون ها فعال هستند. برای همین نیاز است تا برای زمان تست هم scaling انجام شود که پیاده سازی را دشوار می‌کند. یک پیاده سازی موجود دیگر inverted dropout است که فقط برای زمان آموزش است و تغییری برای زمان تست ایجاد نمی‌کند.
برای اطلاعات بیشتر می‌توانید به این لینک مراجعه کنید.

Early Stopping

همان‌طور که میدانید، در overfitting میزان loss برای داده های آموزش کاهش پیدا می‌کند ولی برای داده های validation مقدار loss تابعی افزایشی میشود که در شکل ۲ نیز قابل مشاهده است. تکنیک early stopping به این صورت است که با استفاده از مقایسه loss برای داده های validation سعی بر این داریم تا قبل از overfit شدن، یادگیری شبکه را متوقف کنیم.



شکل ۲: early stopping

در این قسمت شما باید این تکنیک را پیاده سازی کنید. توجه داشته باشید که loss ممکن است تابعی

اکیدا نزولی نباشد در واقعیت و شامل نویز هایی باشد و برای همین در نظر گرفتن اولین زمانی که loss روی داده های validation افزایش پیدا کرد یا ثابت ماند ممکن است نتیجه خوبی ندهد. بهتر است در پیاده سازی، عملکرد validation در چندین epoch را برای early stopping در نظر بگیرید.

تحویل

گزارش خود را به همراه نتایج و کد ها داخل سامانه vu آپلود کنید.