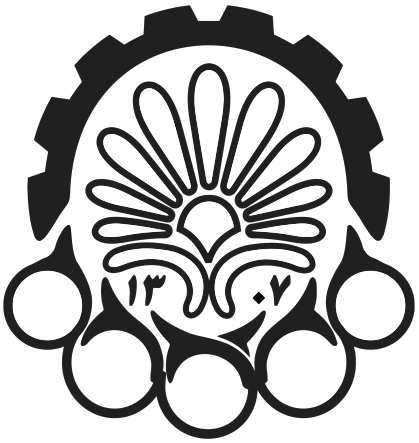
شبکه‌های عصبی و یادگیری عمیق
دکتر صفابخش



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

تمرین دوم

شبکه چندلایه پرسپترونی

۲۱ فروردین ۱۴۰۳

شبکه‌های عصبی و یادگیری عمیق

تمرین دوم

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

دانشکده مهندسی کامپیوتر

سوال اول - عملی

فرض کنید یک مجموعه داده دو کلاسه در اختیار دارید که کاملا به صورت خطی کلاس‌ها از هم جداپذیر هستند. یک شبکه چند لایه پرسپترونی (با طراحی دلخواه) طراحی نموده‌اید که لایه خروجی آن شامل دو نرون می‌باشد که تابع فعال‌ساز `softmax` بر آن اعمال می‌شود. در زمان آموزش، از تابع خطای `binary cross entropy` برای محاسبه خطا و بهینه‌سازی وزن ها استفاده می‌شود. آیا این امکان وجود دارد که خطای حاصل صفر شود؟ اگر امکان ندارد، با استدلال و اثبات ریاضی نشان دهید و اگر امکان دارد، با معرفی چهار داده (دو داده به ازای هر کلاس) و پرسپترون مد نظرتان نشان دهید که خطا می‌تواند دقیقا صفر شود.

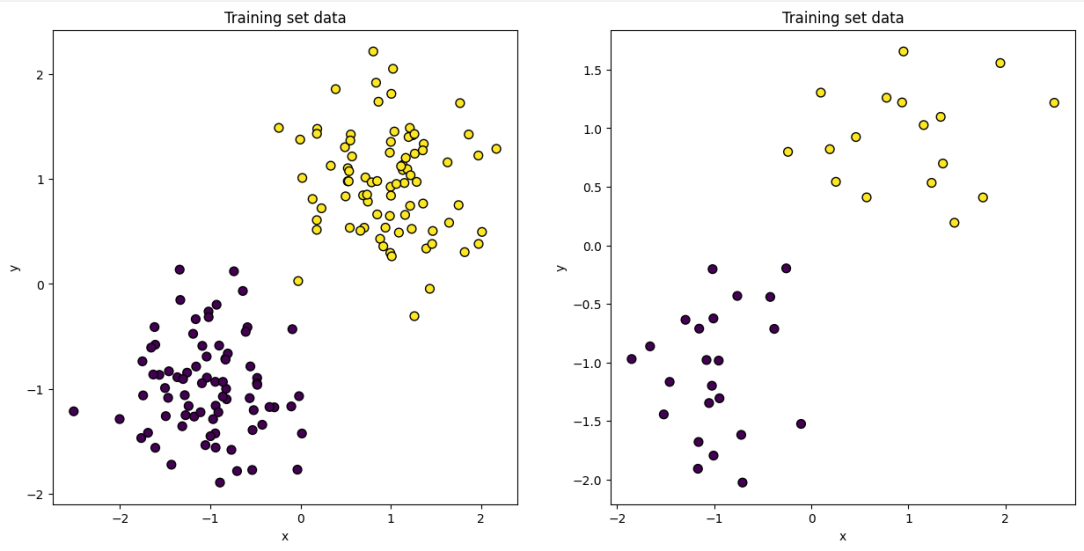
پاسخ

فایل کد از مسیر `code/Q1.ipynb` قابل مشاهده است.

در این سوال برای تولید دیتا از تابع `make_blobs` از کتابخانه `sklearn` به صورت زیر استفاده شده است:

```
x, y = datasets.make_blobs(n_samples=200, centers=[(-1, -1), (1, 1)], cluster_std=0.5)
```

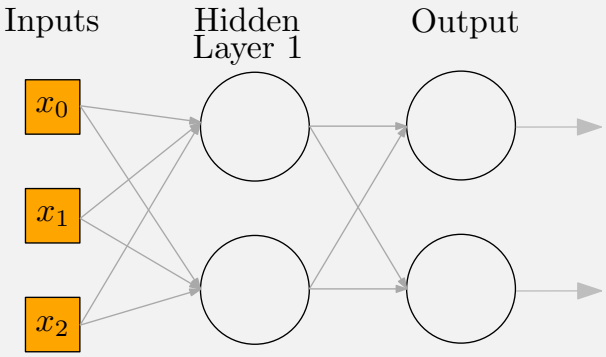
که داده هایی با تعداد ۲۰۰ نمونه و انحراف معیار ۰/۵ حول نقطه های (۱-،۱-) و (۱و۱) می‌کند که کاملا جداپذیر خطی است «شکل ۱»



شکل ۱: داده‌های تولید شده برای مسئله

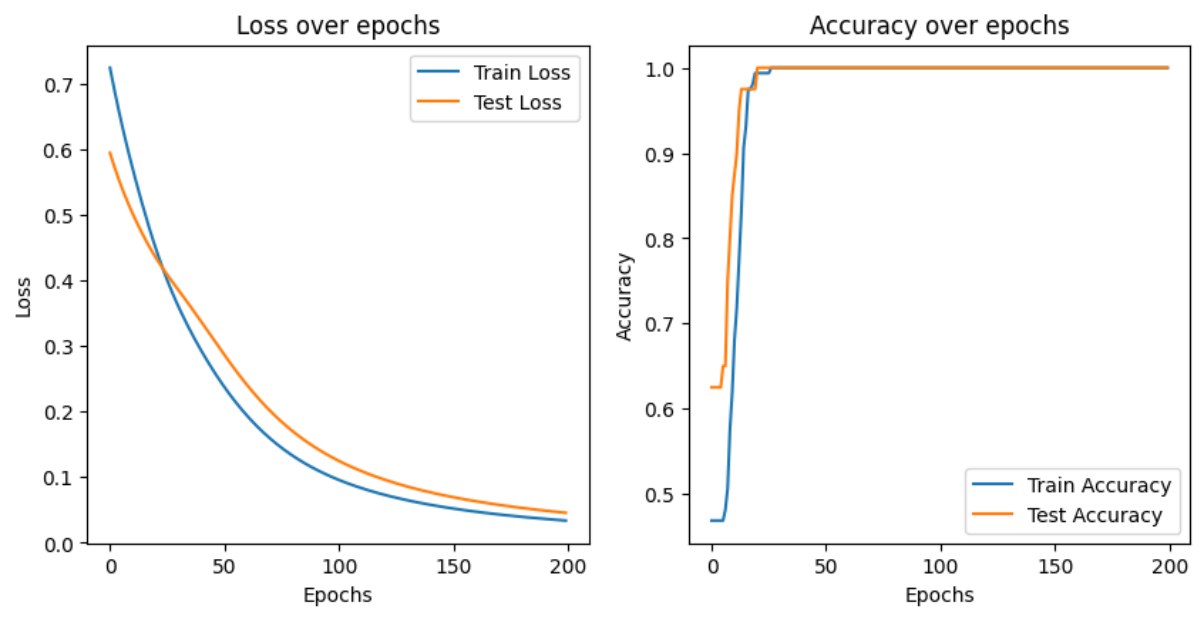
برای حل این مسئله از شبکه‌ی پرسپترونی چند لایه ای با ساختار «شکل ۲» استفاده شده است. با توجه به اینکه داده‌ها جداپذیر خطی هستند، می‌توان این مسئله را با یک نرون پرسپترونی نیز حل نمود اما با توجه به اینکه در صورت سوال گفته شده است شبکه ای چند لایه طراحی کنید، از شبکه چند لایه پرسپترونی استفاده کردیم.

پاسخ



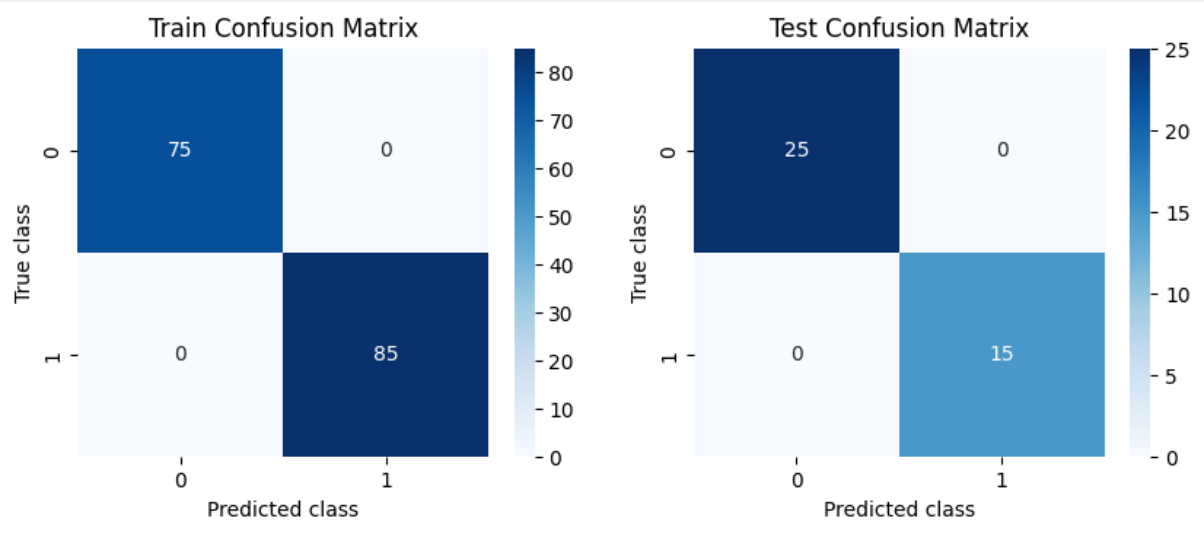
شکل ۲: ساختار شبکه طراحی شده

تابع فعال‌ساز لایه مخفی، ReLU درنظر گرفته شده است و در لایه خروجی نیز از softmax استفاده شده است. طبق خواسته مسئله، از **binary cross entropy** به‌عنوان تابع خطا استفاده شده است. همچنین از تابع بهینه‌ساز ADAM در این مسئله استفاده شده است.
نتایج آموزش در ۲۰۰ دوره آموزشی به صورت زیر گزارش می‌شود:

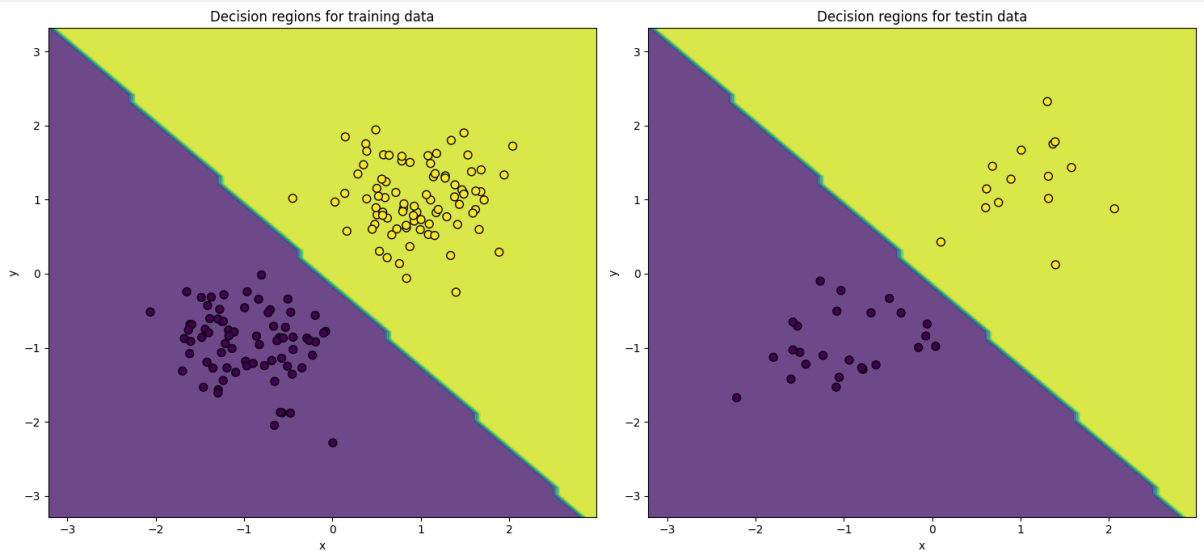


شکل ۳: منحنی خطا و دقت برای داده‌های آموزش و تست

پاسخ



شکل ۴: ماتریس پراکندگی



شکل ۵: ناحیه تصمیم

سوال دوم - نظری

تعیین اندازه دسته (Batch) به چه عواملی بستگی دارد و تاثیر آن در روند آموزش شبکه چیست؟فرض کنید که اندازه و سایز هر نمونه از مجموعه داده بگونه ای بزرگ و حجیم است که وقتی اندازه دسته بیش از ۲ باشد، خطای حافظه دریافت می‌شود. (Out of memory) چگونه می‌تواند این مشکل و چالش را بدون ارتقای سخت‌افزار حل نمود؟ راهکار مدنظر را معرفی و با جزئیات کامل پیاده سازی کنید.

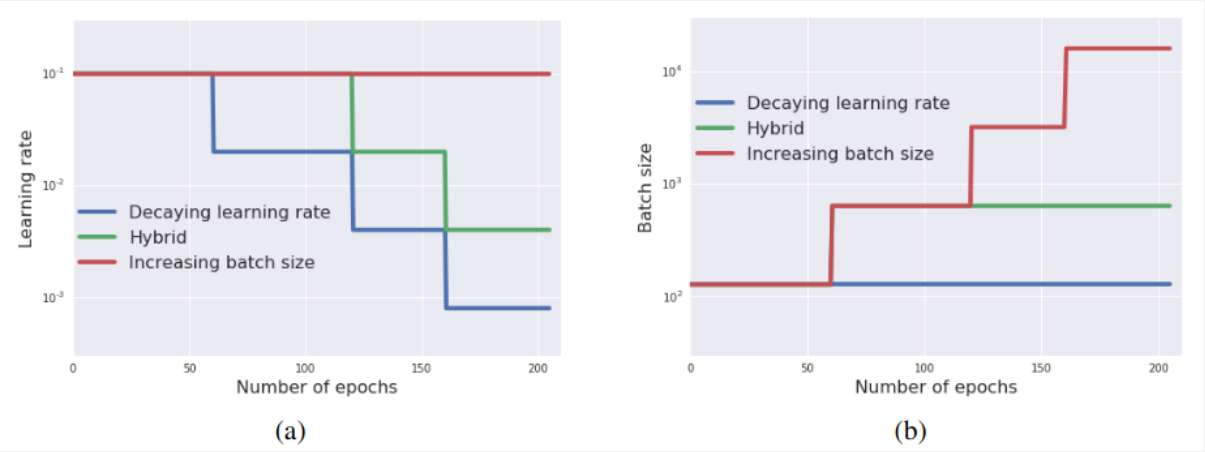
پاسخ

در سیستم‌های یادگیری عمیق، هایپرپارامترهای زیادی دخیل هستند، یکی از مهمترین آنها اندازه دسته یا batch size است. [۱] اندازه دسته از این جهت مهم است که زیرا با افزایش اندازه دسته می‌توان سرعت محاسباتی را برای آموزش مدل افزایش داد. اما مشکلی که اندازه دسته بزرگ می‌تواند ایجاد کند عدم آموزش مناسب شبکه است. زیرا اگر اندازه دسته خیلی بزرگ باشد، در فرایند آموزش، داده هایی که در ابتدای دسته وجود داشتند، هنگامی که فرآیند آموزش به انتهای دسته می‌رسد موجب این می‌شود که داده‌های ابتدایی فراموش شود و شبکه در memorisation دچار مشکل می‌شود و شبکه به درستی آموزش نمی‌بیند. [۲]

همواره مصالحه ای بین اندازه دسته و سرعت همگرایی وجود دارد. اندازه دسته ای برابر با اندازه کل مجموعه داده ها، همگرایی را تضمین می‌کند اما به قیمت کاهش سرعت فرایند آموزش از طرفی کوچک گرفتن اندازه دسته سرعت همگرایی را بالا می‌برد اما تضمینی برای همگرایی مدل وجود ندارد. [۲]

بنابر این پیشنهاد می‌شود که از اندازه دسته کوچک شروع به آموزش دادن شود و در حین آموزش کم کم اندازه دسته را افزایش داد. [۲]

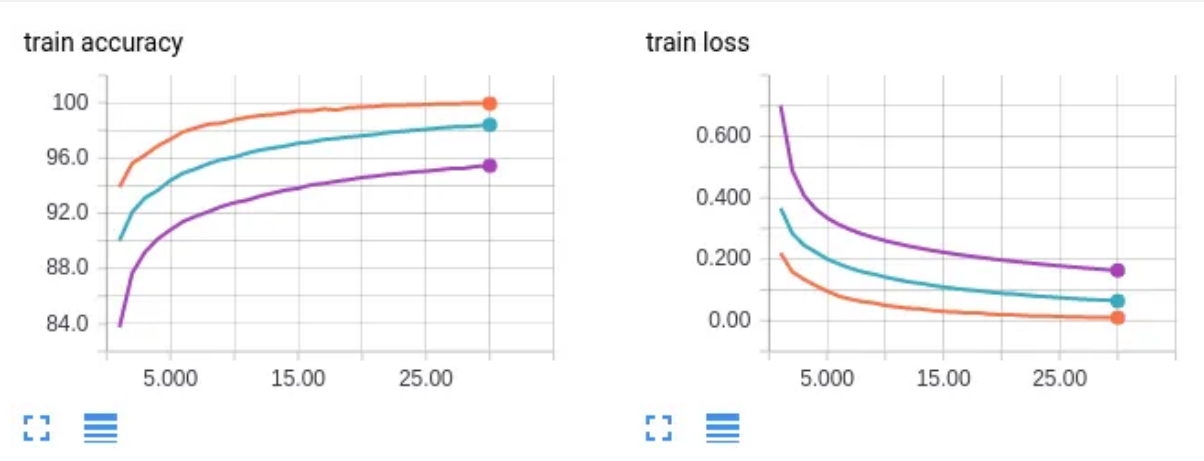
در [۲] روند تغییر هایپرپارامترهای شبکه مثل نرخ یادگیری و اندازه دسته به صورت زیر پیشنهاد شده است:



شکل ۶: روند تغییر پویا هایپرپارامترهای مهم شبکه

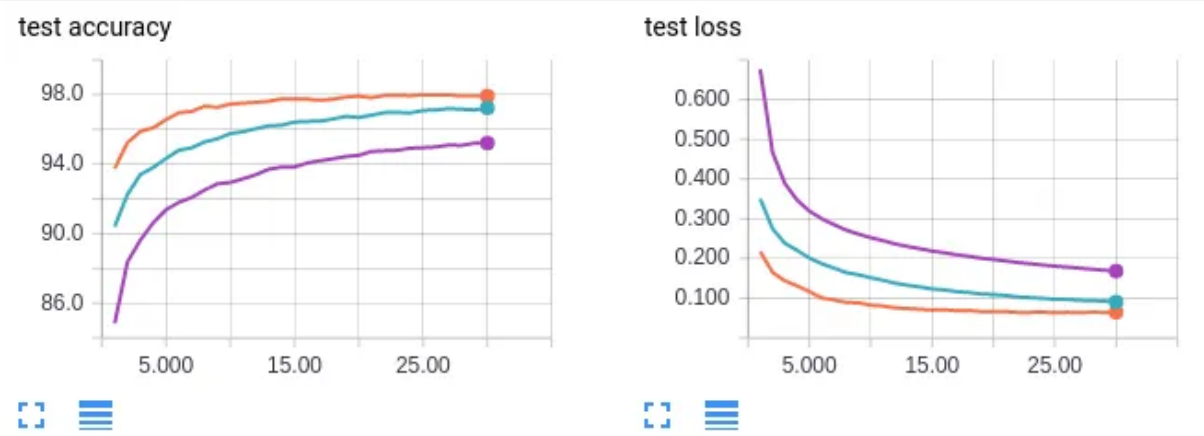
در ادامه مثالی از سرعت همگرا شدن یک شبکه MLP که با اندازه دسته های مختلف بر روی دیتاست MNIST آموزش داده شده است آورده شده است. در شکل های «۷ و ۸» نمودار نارنجی رنگ مربوط به اندازه دسته ۶۴، نمودار آبی مربوط به اندازه دسته ۲۵۶ و نمودار بنفش مربوط به اندازه دسته ۱۰۲۴ است. همانطور که توضیح داده شد، سرعت همگرایی با اندازه دسته کوچکتر، بیشتر است.

پاسخ



شکل ۷: تاثیر اندازه دسته در سرعت همگرایی آموزش

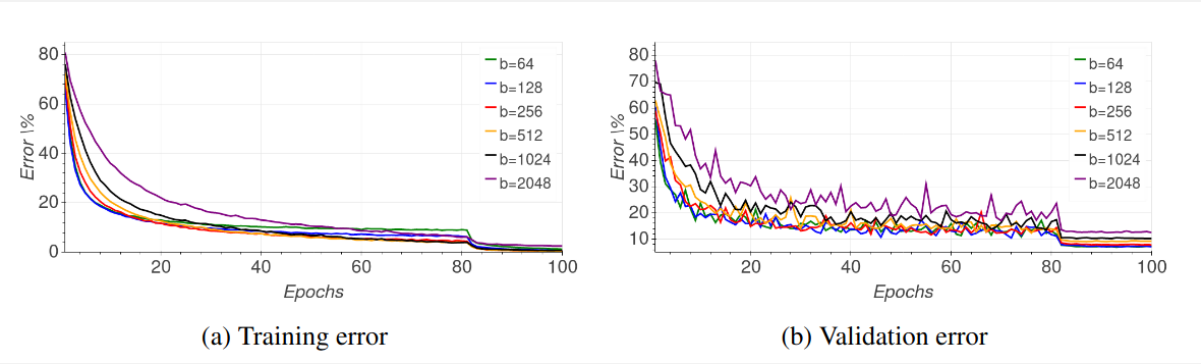
این قضیه در مورد فاز تست شبکه پس از آموزش نیز صادق است:



شکل ۸: تاثیر اندازه دسته در سرعت همگرایی تست شبکه

یکی دیگر از عواملی که در تعیین اندازه دسته مهم است، حافظه سیستمی است که از آن استفاده می‌کنیم. اگر حافظه سیستم‌مان محدود باشد نمی‌توان اندازه دسته را بزرگ گرفت چرا که به هنگام آموزش حافظه سیستم پر خواهد شد و با خطای out of memory مواجه خواهیم شد. بنابر این در پاسخ به قسمت دوم سوال می‌توان گفت در صورت مواجه شدن با این خطا می‌بایست اندازه دسته را کوچک تر کنیم. با تکرار و سعی و خطا می‌توان اندازه دسته مناسب برای آنکه حافظه پر نشود را پیدا کرد.
در شکل «۹» می‌توان مثال دیگری از این قضیه را مشاهده کرد.

پاسخ



(a) Training error (b) Validation error

شکل ۹: تاثیر اندازه دسته در سرعت همگرایی تست شبکه [۳]

*

References

[1] Bengio Y. Practical recommendations for gradient-based training of deep architectures. InNeural networks: Tricks of the trade: Second edition 2012 May (pp. 437-478). Berlin, Heidelberg: Springer Berlin Heidelberg.

[2] Smith SL, Kindermans PJ, Ying C, Le QV. Don't decay the learning rate, increase the batch size. arXiv preprint arXiv:1711.00489. 2017 Nov 1.

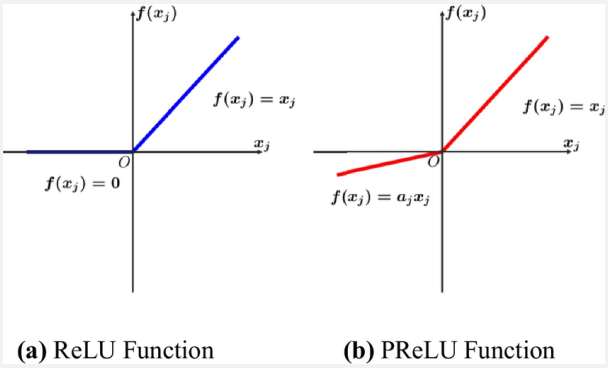
[3] Hoffer E, Hubara I, Soudry D. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. Advances in neural information processing systems. 2017;30.

سوال سوم - نظری

نشان دهید شبکه چند لایه پرسپترونی که فقط از تابع فعال‌سازی ReLU و یا pReLU استفاده می‌کند، تابع پیوسته تکه‌ای خطی می‌سازد.

پاسخ

از شبکه پرسپترون چندلایه می‌توان برای تقریب چندجمله ای های مرتبه پایین تکه ای استفاده کرد. [۱] و [۲] دوتا از توابع غیر خطی‌ای که استفاده می‌شود، تابع ReLU و PReLU است که به صورت زیر تعریف می‌شود:



(a) ReLU Function **(b)** PReLU Function

شکل ۱۰: توابع فعال‌ساز ReLU و PReLU

$$r(x) = max(0, x)$$

بنابراین یک تابع تکه ای خطی را می‌توان به صورت زیر نوشت:

$$f(x) \approx f_l(x) = \sum_{i=0}^{N-1} f(x_i) t_i(x),$$

$$\text{where} \qquad t_i(x) = t(h^{-1}(x - x_i))$$

و تابع $t(x)$ یک تابع مثلثی به صورت زیر است:

$$t(x) = \begin{cases} x + 1, & \text{if } -1 \le x \le 0 \\ -x + 1, & \text{if } 0 \le x \le 1 \\ 0, & \text{otherwise.} \end{cases}$$

با ساختار یک گره در ورودی که $x \in [0, 1]$ و یک گره در خروجی که $f(x)$ را می‌سازد می‌توان یک شبکه MLP که متشکل از ۳ لایه (ورودی + مخفی + خروجی) و ۴ نرون وروی است به صورت زیر پیشنهاد داد:

$$R_{in} \to R_{j,1} : \alpha_{j,1} = h^{-1}, \xi_{j,1} = -h^{-1} x_{j-1}$$

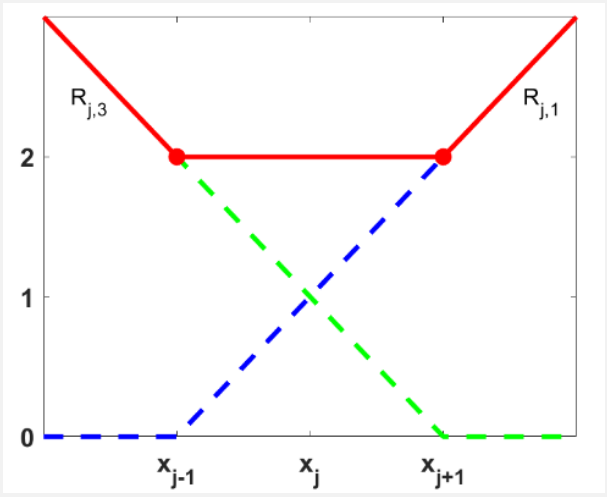
$$R_{in} \to R_{j,2} : \alpha_{j,2} = h^{-1}, \xi_{j,2} = -h^{-1} x_j$$

پاسخ

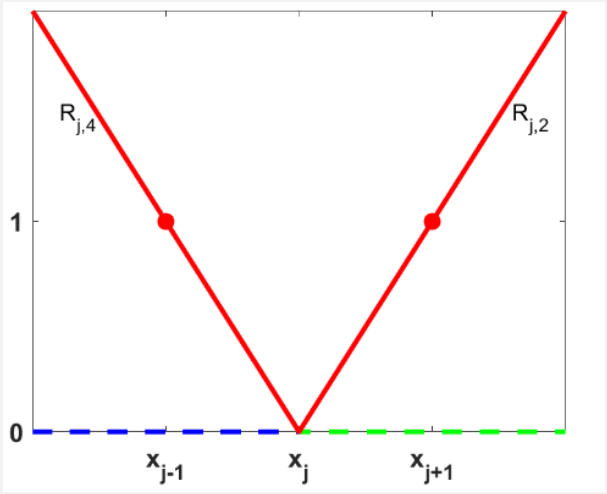
$$R_{in} \to R_{j,3} : \alpha_{j,3} = h^{-1}, \xi_{j,3} = -h^{-1}x_{j+1}$$

$$R_{in} \to R_{j,4} : \alpha_{j,4} = h^{-1}, \xi_{j,4} = -h^{-1}x_{j}$$

در شکل‌های زیر خروجی نرون‌های شبکه پس از عبور از تابع فعال‌سازی ReLU را مشاهده می‌کنیم:



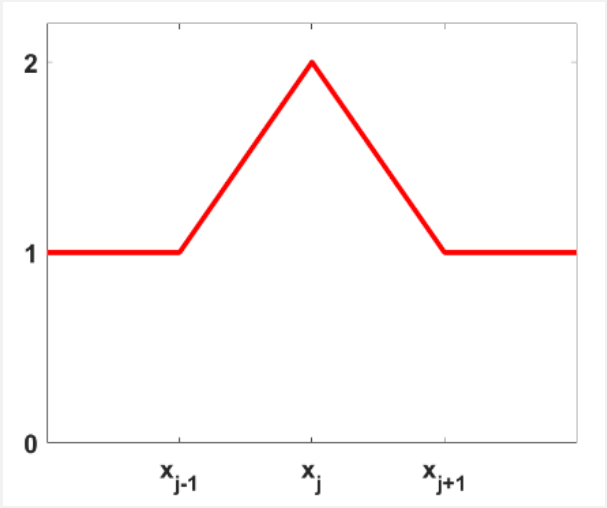
شکل ۱۱: آ



شکل ۱۲: ب

در این دو شکل، نمودار های خط‌چین، خروجی تک نرون هستند و نمودار های صاف، خروجی نرون های ترکیبی.
و در نهایت تابع مثلثی تقریب زده شده با شبکه MLP به صورت زیر می‌شود:

پاسخ



شکل ۱۳: شکل تقریب زده شده

*

References

[1] Lin R, You S, Rao R, Kuo CC. Constructing multilayer perceptrons as piecewise low-order polynomial approximators: a signal processing approach. arXiv preprint arXiv:2010.07871. 2020 Oct 15.

[2] Zeng GL. A deep-network piecewise linear approximation formula. IEEE Access. 2021 Aug 31;9:120665-74.

سوال چهارم - نظری

تقارن در شبکه عصبی به چه معناست؟ آیا نیاز داریم این تقارن را بشکنیم؟ درصورتی که جواب شما مثبت است، کیس مورد نظر را طراحی کرده و توضیح دهید. بررسی کنید چه روش‌های برای شکستن تقارن وجود دارد. چند مورد نام ببرید.

پاسخ

شاید در ابتدا با مطرح کردن مسئله تقارن در شبکه های عصبی، تقارن هندسی شبکه به ذهن آید اما باید گفت که در کنار این تقارن هندسی، تقارن در داده ها و تقارن در الگوریتم نیز وجود دارد. [۱].
تقارن در معماری شبکه بدین معنی است که تعداد نرون های ورودی و خروجی و نرون های مخفی نسبت به هم متقارن باشند.
تقارن در الگوریتم ها نیز یک نقش اساسی در یادگیری ماشین دارد. از آنجایی که الگوریتم های یادگیری ماشین معمولاً پیچیده هستند، استفاده از تقارن در آنها می تواند به تسهیل و بهبود عملکرد آنها کمک کند. به عنوان مثال، الگوریتم های شبکه های عصبی که از تقارن استفاده می کنند، می توانند سریع تر و دقیق تر اطلاعات را پردازش کنند و الگوهای پیچیده تری را متوجه شوند.
تقارن در داده ها این مزیت را دارد که با افزایش دقت و سرعت در تحلیل داده ها و جلوگیری از بروز اشتباهات، در یادگیری کمک می کند. به عنوان مثال، اگر یک تصویر نیمه از یک چهره انسان در دیتاست باشد، تقارن در داده ها می تواند به یادگیری شبکه در تشخیص چهره انسان کمک کند. یادگیری از داده های تقارنی می تواند به شبکه کمک کند تا الگوهای پیچیده تر را بشناسند و بدون تحلیل دقیق تصاویر، به تصمیمات صحیح برسند.
همانند همه پدیده های حاکم بر طبیعت که متقارن هستند، در شبکه‌های عصبی مصنوعی نیز تقارن یک تصل مهم در طراحی است. روش هایی برای شکستن تقارن وجود دارد مانند:

۱. Dropout: در هر مرحله از آموزش به صورت رندم تعدادی از وزن ها را حذف می‌کند
۲. Batch normalization
۳. data Augmentation: با اعمال تبدیلات مختلف به داده‌ها، مانند چرخش، برش، شیفت دادن و تغییر مقیاس، تنوع در داده‌ها افزایش می‌یابد و تقارن‌ها کاهش می‌یابند.

و با دلایل مطرح شده شکستن یا عدم شکستن تقارن در شبکه، بستگی به کاربرد و نوع تقارن دارد. در برخی از کاربرد ها می‌تواند مفید باشد و در برخی خیر.

*

References

[1] Tanaka H, Kunin D. Noether's learning dynamics: Role of symmetry breaking in neural networks. Advances in Neural Information Processing Systems. 2021 Dec 6;34:25646-60.

[2] Kaba SO, Ravanbakhsh S. Symmetry Breaking and Equivariant Neural Networks. arXiv preprint arXiv:2312.09016. 2023 Dec 14.

سوال پنجم - عملی

ابروضوح یک کاربرد در بینایی کامپیوتر می‌باشد که در آن هدف ارتقای وضوح تصاویر می‌باشد. این امر می‌تواند در مقاصد مختلف نظیر تصویر برداری پزشکی، بهبود تصاویر نظارتی–امنیتی، بازسازی تصاویر قدیمی و ... به‌کار گرفته شود. در این سوال هدف طراحی و پیاده سازی یک شبکه عصبی چندلایه برای هدف فوق می‌باشد.

۱. ۱۰ تصویر دلخواه از اینترنت که حاوی گستره رنگی مختلفی می‌باشد را به‌عنوان مجموعه داده انتخاب کنید و آن را نمایش دهید. حال وضوح هر یک از تصاویر را نصف کنید. اکنون به ازای هر یک از پیکسل ها در عکس اصلی، متناظر آن و هشت همسایگی مجاور آن در عکس با وضوح پایین تر را بیابید و مجموعه داده موردنظر را بدست آورید. ابعاد ورودی برابر با ۲۷ ویژگی (پیکسل متناظر و هشت همسایگی آن به ازای سه کانال رنگی در وضوح پایین) خواهد بود و خروجی (لیبل) نیز شامل سه مقدار(مقدار سه کانال RGB در تصویر اصلی) خواهد بود. این روند را برای تمامی پیکسل های ۱۰ تصویر انجام دهید تا برای هر تصویر i یک مجموعه داده به صورت $W_i * H_i, 27, 3$ پدید آید. دو تصویر را برای آزمون، و یک تصویر را برای اعتبار سنجی و هفت تصویر باقی مانده را برای آموزش استفاده کنید. می‌توانید پیکسل های حاصل از تصاویر مختلف در گروه آموزش را باهم ترکیب کرده و درهم (shuffle) سازید که ابعاد آن مجموعه داده به صورت $\sum_{i=1}^{7} W_i * H_i, 27, 3$ درآید.

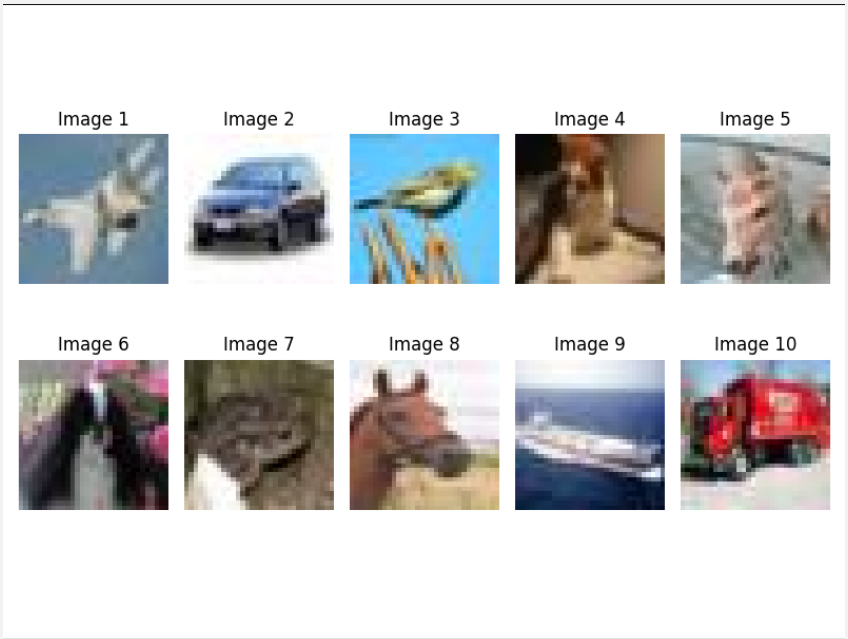
۲. یک شبکه چند لایه پرسپترونی طراحی و آموزش دهید که بتواند به ازای ۲۷ ویژگی ورودی در وضوح پایین، مقدار پیکسل رنگی در وضوح بالا را محاسبه کنید. معماری خود را ترسیم نموده و آموزش شبکه را توضیح دهید. از چه تابع خطایی برای آموزش استفاده کرده‌اید؟ موارد ذکر شده در ابتدای پروژه را برای این سوال به‌صورت کامل گزارش دهید و نتایج را تحلیل کنید.

۳. مقدار تابع خطا را به ازای مجموعه داده آزمون محاسبه و گزارش نمایید و در گام نهایی تصاویر با وضوح بالا را تولید نموده و آن را با تصاویر اصلی مقایسه کنید. علاوه بر مقایسه‌ی بصری، معیارهای کمی SSIM و PSNR را مطالعه کرده و بر اساس آن عملکرد شبکه خود را ارزیابی کنید.

۴. اگر از شبکه‌ی قسمت ۲، دوبار متوالی استفاده شود، می‌تواند وضوح تصویر را چهار برابر کند. به نظر شما این رویکرد می‌تواند مفید واقع شود یا یک شبکه ای که به صورت مستقیم چهار برابر وضوح را افزایش می‌دهد؟ با انجام آزمایش و گزارش کمی نتیجه مورد نظر را نشان دهید.

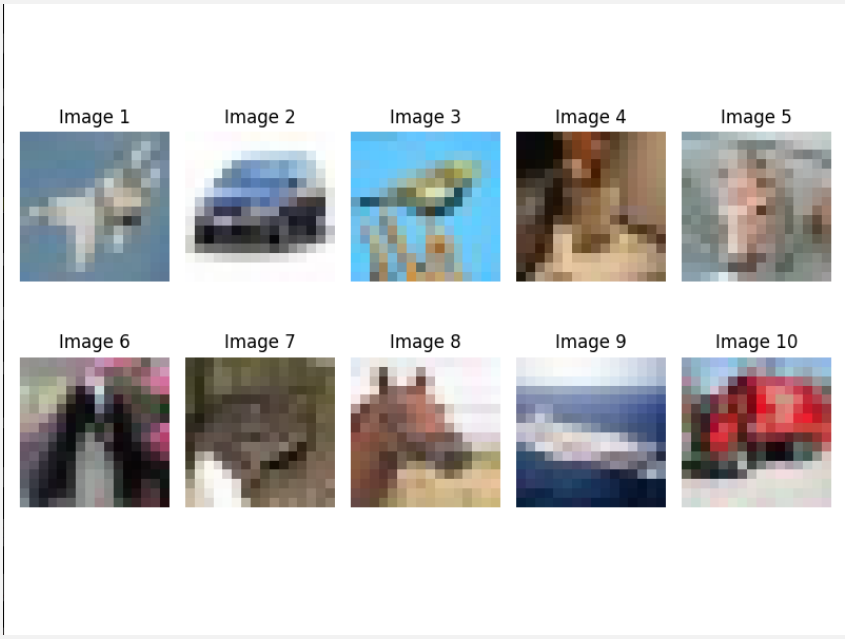
پاسخ

فایل کد از مسیر `code/Q1.ipynb` قابل مشاهده است.
دیتاست مورد استفاده در این تمرین، دیتاست CIFAR10 است که ۱۰ نمونه از آن به عنوان ورودی شبکه انتخاب شده است.



شکل ۱۴: تصاویر اصلی مورد استفاده در این سوال

سپس طبق صورت سوال وضوح تصاویر را تصف می‌کنیم:



شکل ۱۵: تصاویر با وضوح نصف

پاسخ

سپس ویژگی های گفته شده را از تصویر استخراج می‌کنیم. خروجی و ابعاد ویژگی‌های استخراج شده به صورت زیر است:

```
Size of features tensor: torch.Size([10240, 27])
Size of labels tensor: torch.Size([10240, 3])
```

همانطور که در صورت سوال نیز گفته شده است ابعاد بردار ویژگی‌ها باید به صورت (۳، ۲۷، ۱۰۲۴) باشد. در ابعاد گزارشی توسط ما عدد ۱۰۲۴ به ۱۰۲۴۰ تبدیل شده است که این به دلیل آن است که ۱۰ عدد تصویر داشتیم. در این کد ما قسمت لیبل را از بردار ویژگی ها جدا کردیم و خود برداری با ابعاد (۳، ۱۰۲۴۰) شده است.

پس از استخراج ویژگی ها، طبق خواسته مسئله داده ها را به سه دسته آموزش، تست و اعتبارسنجی تقسیم می‌کنیم. ابعاد خروجی به صورت زیر می‌شود:

```
Size of training features tensor: torch.Size([7168, 27])
Size of training labels tensor: torch.Size([7168, 3])
Size of testing features tensor: torch.Size([2048, 27])
Size of testing labels tensor: torch.Size([2048, 3])
Size of validation features tensor: torch.Size([1024, 27])
Size of validation labels tensor: torch.Size([1024, 3])
```

پس از استخراج ویژگی ها نوبت به آموزش شبکه می‌رسد. در این مسال شبکه ای با معماری زیر طراحی شده است:

۱. تعداد لایه ها: ۴ لایه (ورودی + ۲ لایه مخفی + خروجی)

۲. تعداد نرون های ورودی: ۲۷

۳. تعداد نرونهای لایه مخفی اول: ۶۴

۴. تعداد نرون های لایه مخفی دوم: ۱۲۸

۵. تعداد نرون های خروجی: ۳

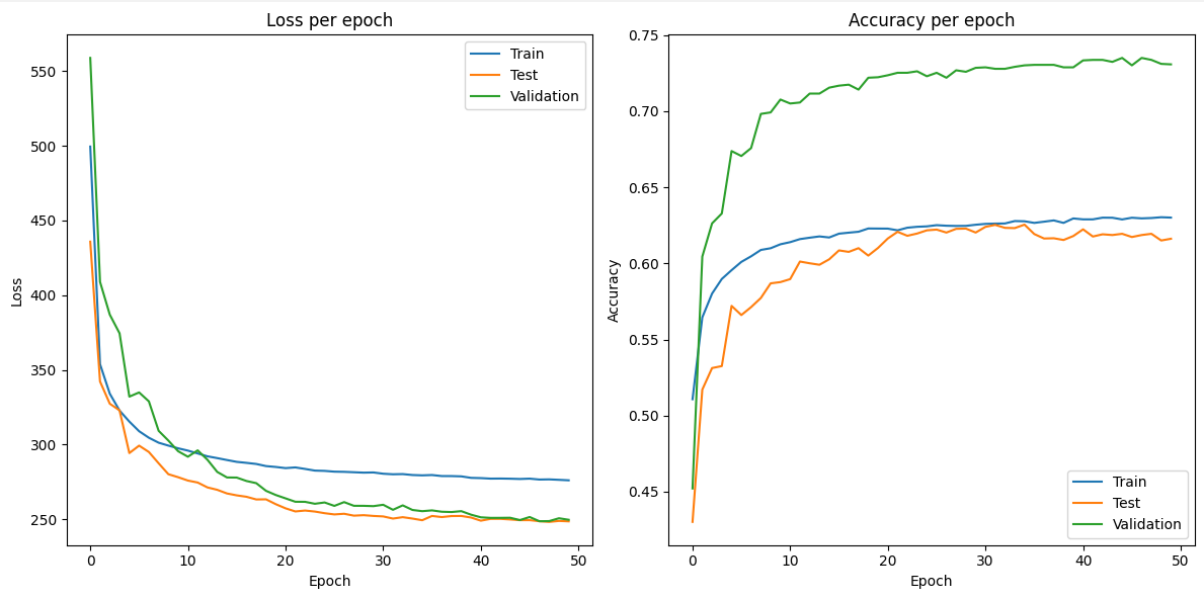
۶. تابع فعال‌ساز همه لایه ها: ReLU

۷. تابع خطا: میانگین مربع خطا

۸. بهینه‌ساز: Adam

نمودار های خطا و دقت بر حسب epoch برای داده‌های آموزش، تست و اعتبارسنجی در ۵۰ دوره، به صورت زیر شده است:

پاسخ



شکل ۱۶: نمودار خطا و دقت

پارامترهای SSIM و PSNR به صورت زیر محاسبه شده است:

1. `Average SSIM: 0.1167`
2. `Average PSNR: 7.0965`

متاسفانه تصاویر به درستی بازسازی نشده اند. خروجی آن در فایل کد موجود است.

فایل تمامی کد ها پیوست شده است. همچنین می‌توانید آن‌ها را از Repository زیر هم دریافت کنید:
github.com/rezaAdinepour/Deep-Learning-Homework