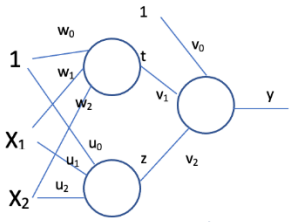
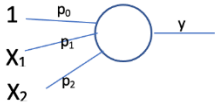
سوال ۱. به سوال‌های زیر درباره‌ی توابع فعالیت پاسخ دهید.

الف) با ذکر مثال و انجام محاسبات توضیح دهید که در صورت عدم استفاده از توابع فعالیت و یا استفاده از توابع فعالیت خطی برای همه‌ی لایه‌ها در یک شبکه‌ی پرسپترون چند لایه، چه اتفاقی می‌افتد.

در صورتی که تابع فعالیت نداشته باشیم و یا برای تمامی لایه‌ها از توابع فعالیت خطی استفاده کنیم، ترکیب خطیِ ترکیب‌های خطی حاصل، در نهایت برابر با یک ترکیب خطی جدید می‌شود و این به این معناست که انگار در عمل، تنها یک پرسپترون وجود دارد و نمی‌توان به معنای واقعی یک شبکه از پرسپترون‌ها داشت. این مساله را در مثال زیر مشاهده می‌کنیم:



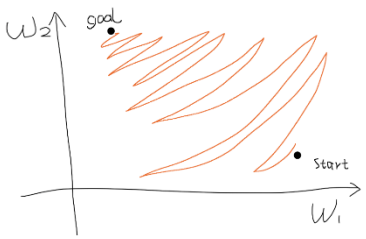
$$t = w_0 + x_1w_1 + x_2w_2 \qquad z = u_0 + x_1u_1 + x_2u_2$$
$$y = v_0 + tv_1 + zv_2$$
$$y = v_0 + (w_0 + x_1w_1 + x_2w_2)v_1 + (u_0 + x_1u_1 + x_2u_2)v_2$$
$$y = v_0 + w_0v_1 + u_0v_2 + (w_1v_1 + u_1v_2)x_1 + (w_2v_1 + u_2v_2)x_2$$
$$y = p_0 + x_1p_1 + x_2p_2$$



ب) تابع فعالیت سیگموید چه مشکلاتی دارد؟

تابع سیگموید به صورت $\sigma(x) = 1/(1+e^{-x})$ و مشتق آن به صورت $\sigma'(x)=\sigma(x)(1-\sigma(x))$ تعریف می‌شود و بنابراین مقادیر بین ۰ و ۱ را اختیار می‌کند. همچنین می‌دانیم که گرادیان این تابع به ازای مقادیر نزدیک به مثبت و منفی بی‌نهایت به صفر میل می‌کند. مشکل عمده‌ی این تابع (و نیز تابع تانژانت هایپربولیک) به خصوص در شبکه‌های عمیق، vanishing gradient است؛ به این معنا که در هنگام backpropagation ، هنگامی که به تدریج از لایه‌های پنهان نزدیک به خروجی به سمت لایه‌های پنهان نزدیک به ورودی حرکت می‌کنیم، در اثر ضرب گرادیان‌های بین صفر و یک (مشتق‌های زنجیره‌ای را به یاد بیاورید)، مقادیر گرادیان بسیار کوچک و نزدیک به صفر می‌شود و این امر باعث می‌شود که وزن‌ها (خصوصا در لایه‌های ابتدایی) به سختی تغییر کنند و بنابراین یادگیری بسیار بسیار کند شود.

اشکال دیگر سیگموید که نسبت به مورد قبلی اهمیت کمتری دارد، این است که خروجی‌های این تابع اصطلاحا non-zero centered هستند که این مطلب روی نحوه‌ی آپدیت شدن وزن‌ها و در نتیجه حرکت به سمت جواب تاثیر می‌گذارد. هنگامی که داده‌ای که به یک نورون وارد می‌شود مثبت است، همه‌ی گرادیان‌ها نسبت به وزن‌ها با یکدیگر هم‌علامت می‌شوند. اگر فرض کنیم که دو وزن داریم، در این حالت گرادیان نسبت به این دو یا هر دو مثبت است و یا هر دو منفی و بنابراین در صفحه یا می‌توانیم به سمت شمال شرق حرکت کنیم یا جنوب غرب. حال اگر فرض کنیم که نقطه بهینه در شمال غرب باشد، حرکت ما به سمت این نقطه زیگ زاگی می‌شود. این حرکات زیگ زاگ معمولا بهینه‌سازی را دشوار می‌کنند.



برای درک بهتر این مورد می‌توانید به لینک زیر مراجعه کنید:

https://stats.stackexchange.com/questions/237169/why-are-non-zero-centered-activation-functions-a-problem-in-backpropagation

پ) تابع فعالیت رلو چه برتری‌هایی نسبت به سیگموید دارد؟

۱- تابع رلو مشکلات vanishing gradient و اشباع شدن را ندارد.

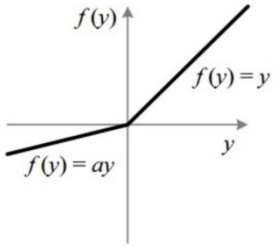
۲- از نظر هزینه‌ی محاسباتی نسبت به سیگموید کاراتر است زیرا برای انجام محاسبات آن تنها به یک عملیات ماکسیمم گیری نیاز است.

۳- در عمل شبکه‌هایی که از رلو استفاده می‌کنند در بیشتر موارد همگرایی بهتر و سریع‌تری دارند.

ت) تابع فعالیت لیکی رلو چگونه تعریف می‌شود و کدام مشکل رلو را حل می‌کند؟

یکی از مشکلات تابع رلو Dying ReLu است که به این علت رخ می‌دهد که تابع رلو به ازای مقادیر ورودی منفی، مقدار خروجی صفر دارد. اگر خروجی یک نورون در مقادیر منفی گیر کند، خروجی نهایی تابع فعالیت (یعنی رلو) همواره برابر با صفر می‌شود و چون شیب رلو در مقادیر منفی صفر است، معمولا اگر نورون در مقادیر منفی گیر کند از این حالت خارج نمی‌شود. به چنین نورونی، که عملا دیگر نقشی در شبکه ایفا نمی‌کند، نورون مرده می‌گویند. در گذر زمان در gradient descent، ممکن است بخشی از نورون‌های شبکه بمیرند. مشکل dying ReLu معمولا وقتی نرخ یادگیری خیلی زیاد است و یا یک bias منفی بزرگ در شبکه وجود دارد، رخ می‌دهد.

تابع Leaky ReLu مانند تابع ReLu تعریف می‌شود، با این تفاوت که برای مقادیر منفی، یک شیب بسیار کوچک در نظر گرفته می‌شود و به این ترتیب مشکل Dying ReLu تا حد زیادی برطرف می‌شود.



ث) لیکی رلو چه کاستی نسبت به رلو دارد؟

در وهله‌ی اول، محاسبات Leaky ReLu به علت اضافه شدن شیب در سمت منفی، کمی از ReLu بیشتر است.
همچنین تنظیم شیب قسمت منفی (که یک پارامتر اضافه نسبت به ReLu است و قبل از شروع آموزش باید انجام شود) برای بهینه کردن عملکرد مدل در Leaky Relu اهمیت می‌یابد. در Parametric ReLu مقدار شیب نیز یاد گرفته می‌شود که می‌توانید در مورد آن بیشتر مطالعه کنید.

سوال ۲. به سوال‌های زیر درباره‌ی بیش برازش و پیش برازش پاسخ دهید.

الف) بیش برازش و پیش برازش را در شبکه‌های عصبی توضیح دهید.

بیش برازش : در این حالت مدل ما به خوبی به داده های آموزشی fit شده است ولی با داده های جدید و تست نمیتواند خود را تطبیق دهد. این حالت معمولا وقتی رخ میدهد که تعداد پارامترها زیاد است و مدل زیاد آموزش دیده است.

پیش برازش: این حالت وقتی رخ میدهد که مدل بسیار ساده باشد و حتی نتواند خود را به داده های آموزش fit کند. در واقع در این حالت الگوریتم یک مدل خیلی کلی از مجموعه آموزشی به دست میاورد که خطای بسیار قابل توجهی دارد.

ب) در چه سناریوهایی ممکن است بیش برازش یا پیش برازش رخ دهد؟ برای هر یک مثال بزنید.

مثال سناریو بیش برازش :

- زمانی که الگوریتم یادگیری ماشین از مجموعه داده های آموزشی بسیار بزرگتری در مقایسه با مجموعه آزمایشی استفاده می کند و الگوهایی را در فضای ورودی بزرگ می آموزد که فقط دقت را در یک مجموعه آزمایش کوچک به حداقل می رساند.
- زمانی که الگوریتم یادگیری ماشین از پارامترهای زیادی برای مدل سازی داده های آموزشی استفاده می کند.

مثال سناریو پیش برازش :

- هنگامی که مجموعه آموزشی مشاهدات بسیار کمتری نسبت به متغیرها دارد، ممکن است منجر به مدل‌های یادگیری ماشینی با سوگیری کم یا عدم تناسب شود. در چنین مواردی، الگوریتم یادگیری ماشین نمی تواند هیچ رابطه ای بین داده های ورودی و متغیر خروجی پیدا کند زیرا الگوریتم یادگیری ماشین برای مدل سازی داده ها به اندازه کافی پیچیده نیست.
- اگر تعداد ورودی ها زیاد و بعد داده بالا باشد و مدل به اندازه کافی پیچیده نباشد یا قدرت محاسباتی الگوریتم پایین باشد.

پ) چگونه می‌توان مشکل پیش برازش را برطرف کرد؟

برای حل مشکل پیش برازش راهکارهای زیر ارائه میشود :

- پیچیدگی مدل را زیاد کنیم
- تعداد featureها را زیاد کنیم
- تعداد epochهای یادگیری را زیاد کنیم
- نویزها را از داده ها حذف کنیم
- داده های آموزشی را افزایش دهیم.

ت) از روش‌های برطرف کردن بیش برازش، Dropout و Regularization را توضیح دهید

:Dropout

در این روش برای همه نورون ها به غیر از نورون های آخر یک عدد تصادفی تولید میکنیم. آن نورون هایی که عدد تصادفی آنها کمتر از ۰.۵ است را علامت گذاری کرده و بعد تمام وزنهای ورودی و خروجی به آنها را حذف میکنیم. با این کار نقش نورون های بلا استفاده را حذف کرده و شبکه را سبکتر میکنیم. در نتیجه منحنی تولید شده پیچیده نیست و بیش برازش رخ نمیدهد.

Regularization:

روش Regularization ، راهی برای پیدا کردن یک bias-variance tradeoff خوب هست که با تنظیم پیچیدگی مدل اتفاق میفتد.رایج ترین شکل regularization ، به اصطلاح L2 regularization نامیده میشود اما نسخه L1 آن نیز موجود است که فرمول‌های آن به شرح زیر است:

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

لاندا پارامتری است که وابسته به شرایط می تواند تنظیم شود یعنی مقدار بالای وزن ها با درنظر گرفتن مقدار بالایی برای لاندا قابل کنترل خواهد بود و بطور مشابه مقدار کم برای لاندا به منظور تنظیم مقدار کم وزن ها درنظر گرفته می شود. از آنجایی که تابع هزینه باید حداقل رسانی شود، تناسب مطرح شده منطقی به نظر می رسد و با اضافه کردن مجذور نرمال ماتریس وزن ها و ضرب آن در پارامتر نظم دهی، وزن های زیاد به نوعی تنظیم می شود و تابع هزینه کاهش می یابد.

سوال ۳. یک شبکه‌ی عصبی fully connected را در نظر بگیرید که دو ورودی می‌گیرد و دو لایه‌ی پنهان دارد. لایه‌ی ورودی، دو نورون با نام‌های n_1 و n_2 و لایه‌ی پنهان اول دو نورون با نام‌های n_3 و n_4 دارد که به ترتیب دارای بایاس‌های b_3 و b_4 هستند. لایه‌ی پنهان دوم نیز سه نورون با نام‌های n_5، n_6 و n_7 و بایاس‌های به ترتیب b_5، b_6 و b_7 دارد. در نهایت لایه‌ی خروجی نیز شامل یک نورون با نام n_8 و بایاس b_8 است. همه‌ی نورون‌ها (به جز نورون‌های لایه‌ی اول) از تابع فعالیت سیگموید استفاده می‌کنند و تابع هزینه به صورت زیر تعریف می‌شود که در آن y خروجی شبکه است:

$$cost = (y - y^*)2$$

اگر وزن بین نورون‌های n_x و n_y با w_{xy} نمایش داده شود شکل این شبکه‌ی عصبی را رسم کنید و به سوال‌های زیر پاسخ دهید.

الف) مشتق هزینه نسبت به w_{14} را به دست بیاورید (راهنمایی: برای اینکار از قاعده‌ی زنجیره‌ای استفاده کنید).

ب) اگر مقادیر وزن‌ها و بایاس‌ها به صورت زیر باشند:

$w_{13} = -2$ $w_{14} = 4$ $w_{23} = 3$ $w_{24} = -1$

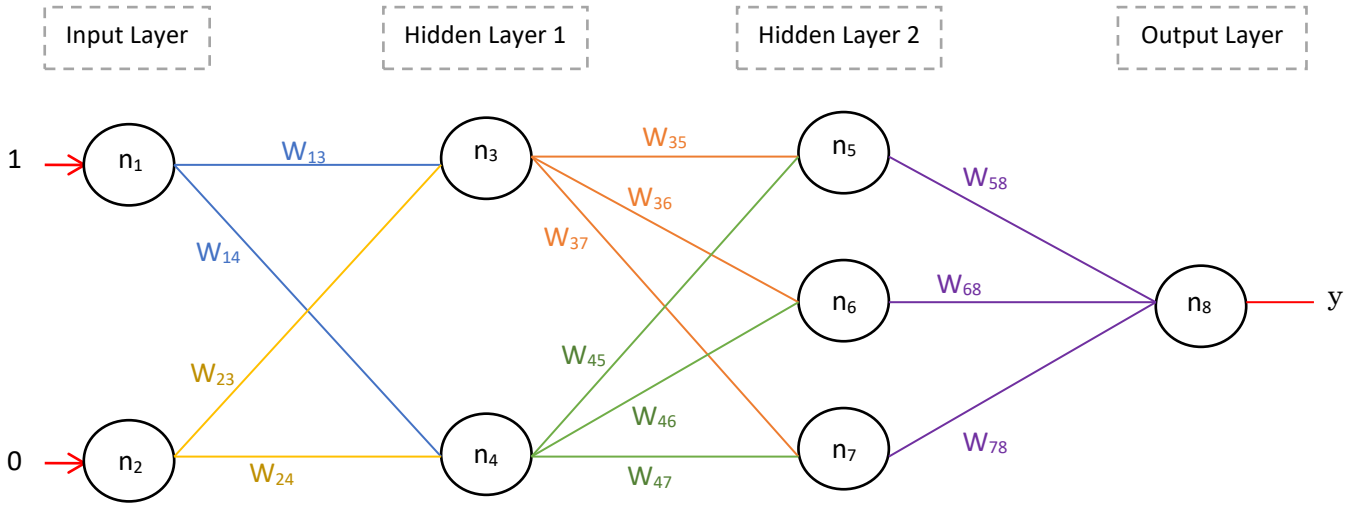
$w_{35} = 1$ $w_{36} = -1$ $w_{37} = -1$ $w_{45} = -1$ $w_{46} = 1$ $w_{47} = 2$

$w_{58} = 2$ $w_{68} = 4$ $w_{78} = 1$

$b_3 = 0.4$ $b_4 = 0.5$ $b_5 = 0.4$ $b_6 = 0.1$ $b_7 = 1$ $b_8 = 0.7$

خروجی شبکه‌ی عصبی را با مقادیر ورودی زیر به دست آورید و هزینه را محاسبه کنید (تمام مراحل محاسبه‌ی خروجی ذکر شود و y^* را برابر با صفر در نظر بگیرید).

$n_2 = 0$ $n_1 = 1$



الف)

بایاس x تابع فعالیت x وزن = فرمول کلی

$$z_1 = b_8 + n_5w_{58} + n_6w_{68} + n_7w_{78}$$

$$z_2 = b_5 + n_3w_{35} + n_4w_{45}$$

$$z_3 = b_6 + n_3w_{36} + n_4w_{46}$$

نوشتن این موارد اجباری نیست اما به حل مسئله کمک می‌کند.

$$z_4 = b_7 + n_3 w_{37} + n_4 w_{47}$$

$$z_5 = b_3 + n_1 w_{13} + n_2 w_{23}$$

$$z_6 = b_4 + n_1 w_{14} + n_2 w_{24}$$

$$\frac{\partial cost}{\partial y} \times \frac{\partial y}{\partial z_1} \times \frac{\partial z_1}{\partial n_5} \times \frac{\partial n_5}{\partial z_2} \times \frac{\partial z_2}{\partial n_4} \times \frac{\partial n_4}{\partial z_6} \times \frac{\partial z_6}{\partial w_{14}}$$

$$\frac{\partial cost}{\partial y} \times \frac{\partial y}{\partial z_1} \times \frac{\partial z_1}{\partial n_6} \times \frac{\partial n_6}{\partial z_3} \times \frac{\partial z_3}{\partial n_4} \times \frac{\partial n_4}{\partial z_6} \times \frac{\partial z_6}{\partial w_{14}}$$

$$\frac{\partial cost}{\partial y} \times \frac{\partial y}{\partial z_1} \times \frac{\partial z_1}{\partial n_7} \times \frac{\partial n_7}{\partial z_4} \times \frac{\partial z_4}{\partial n_4} \times \frac{\partial n_4}{\partial z_6} \times \frac{\partial z_6}{\partial w_{14}}$$

تمامی این موارد را با هم جمع میزنیم.

$$\frac{\partial cost}{\partial w_{14}} = \frac{\partial cost}{\partial y} \times \frac{\partial y}{\partial z_1} \times \frac{\partial n_4}{\partial z_6} \times \frac{\partial z_6}{\partial w_{14}} \left(\left(\frac{\partial z_1}{\partial n_5} \times \frac{\partial n_5}{\partial z_2} \times \frac{\partial z_2}{\partial n_4} \right) + \left(\frac{\partial z_1}{\partial n_6} \times \frac{\partial n_6}{\partial z_3} \times \frac{\partial z_3}{\partial n_4} \right) + \left(\frac{\partial z_1}{\partial n_7} \times \frac{\partial n_7}{\partial z_4} \times \frac{\partial z_4}{\partial n_4} \right) \right)$$

هر کسر را به صورت جداگانه حساب می‌کنیم:

$$\frac{\partial cost}{\partial y} = 2(y - y^*)$$

$$\frac{\partial y}{\partial z_1} = sigmoid(z_1)\left(1 - sigmoid(z_1)\right)$$

$$\frac{\partial n_4}{\partial z_6} = sigmoid(z_6)\left(1 - sigmoid(z_6)\right)$$

$$\frac{\partial z_6}{\partial w_{14}} = n_1$$

$$\frac{\partial z_1}{\partial n_5} \times \frac{\partial n_5}{\partial z_2} \times \frac{\partial z_2}{\partial n_4} = w_{58} \times sigmoid(z_2)\left(1 - sigmoid(z_2)\right) \times w_{45}$$

$$\frac{\partial z_1}{\partial n_6} \times \frac{\partial n_6}{\partial z_3} \times \frac{\partial z_3}{\partial n_4} = w_{68} \times sigmoid(z_3)\left(1 - sigmoid(z_3)\right) \times w_{46}$$

$$\frac{\partial z_1}{\partial n_7} \times \frac{\partial n_7}{\partial z_4} \times \frac{\partial z_4}{\partial n_4} = w_{78} \times sigmoid(z_4)\left(1 - sigmoid(z_4)\right) \times w_{47}$$

جواب آخر به صورت زیر خواهد بود:

$$\frac{\partial cost}{\partial w_{14}} = 2(y - y^*) \times sigmoid(z_1)\left(1 - sigmoid(z_1)\right) \times sigmoid(z_6)\left(1 - sigmoid(z_6)\right) \times n_1 \times ((w_{58} \times sigmoid(z_2)\left(1 - sigmoid(z_2)\right) \times w_{45}) + (w_{68} \times sigmoid(z_3)\left(1 - sigmoid(z_3)\right) \times w_{46}) + (w_{78} \times sigmoid(z_4)\left(1 - sigmoid(z_4)\right) \times w_{47}))$$

$$\frac{\partial cost}{\partial w_{14}} = 2(y - y^*) \times sigmoid(z_1)\left(1 - sigmoid(z_1)\right) \times w_{58} \times sigmoid(z_2)\left(1 - sigmoid(z_2)\right) \times w_{46} \times sigmoid(z_6)\left(1 - sigmoid(z_6)\right) \times n_1$$

ب)

$$n_3 = sigmoid(z_5) = sigmoid(b_3 + n_1 w_{13} + n_2 w_{23}) = sigmoid(0.4 - 2) = 0.167$$

$$n_4 = sigmoid(z_6) = sigmoid(b_4 + n_1 w_{14} + n_2 w_{24}) = sigmoid(0.5 + 4) = 0.989$$

$$n_5 = sigmoid(z_2) = sigmoid(b_5 + n_3 w_{35} + n_4 w_{45}) = sigmoid(0.4 + 0.167 - 0.989) = 0.396$$

$$n_6 = sigmoid(z_3) = sigmoid(b_6 + n_3 w_{36} + n_4 w_{46}) = sigmoid(0.1 - 0.167 + 0.989) = 0.715$$

$$n_7 = sigmoid(z_4) = sigmoid(b_7 + n_3 w_{37} + n_4 w_{47}) = sigmoid(0.1 - 0.167 + 1.978) = 0.871$$

$$y = sigmoid(z_1) = sigmoid(b_8 + n_5 w_{58} + n_6 w_{68} + n_7 w_{78}) = sigmoid(0.7 + 0.792 + 2.86 + 0.871) = 0.994$$

سوال ۴. به سوال‌های زیر درباره‌ی شبکه‌های توابع پایه‌ی شعاعی پاسخ دهید.

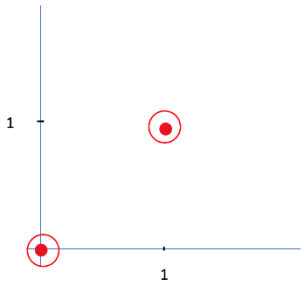
الف) شبکه‌های RBF و MLP را از نظر تعداد لایه‌های پنهان، زمان لازم برای آموزش، زمان عملکرد، سادگی تفسیر معنا و تاثیر هر لایه و یا پرسپترون و در نهایت حساسیت نسبت به نویز مقایسه کنید.

	RBF	MLP
Number of Hidden Layers	یک	یک یا بیشتر
Training Time	کمتر (سرعت بیشتر)	بیشتر (سرعت کمتر)
Functioning Time	بیشتر (سرعت کمتر)	کمتر (سرعت بیشتر)
Interpretability	بیشتر	کمتر
Sensitivity to noise	بیشتر	کمتر

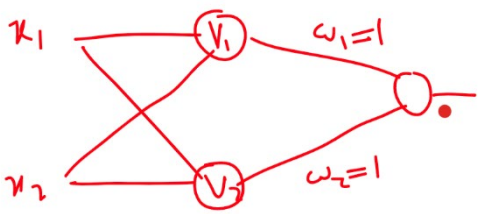
ب) با استفاده از توابع پایه‌ی شعاعی و فرضیات مناسب، تابع xnor را مدل کنید و برای هر یک از چهار حالت ورودی‌های x_1 و x_2 درستی مدل را نشان دهید (توضیحات کافی را برای هر یک از مراحل کار ارائه کنید).

X_1	X_2
0	0
0	1
1	0
1	1

برای مدل کردن xnor به دو RBF نیاز داریم. ابتدا مراکز آنها را تعیین می‌کنیم. مطابق شکل روبه‌رو (محور افقی نشان‌دهنده‌ی x1 و محور عمودی x2 است)، مراکز را در نقاط [۱،۱] و [۰،۰] در نظر می‌گیریم. می‌دانیم در RBF، $\mu = e^{-\gamma\|x-v\|^2}$ است. برای اینکه شعاع‌ها کوچک شوند باید γ را بزرگ در نظر بگیریم (برای مثال آن را در اینجا ۱۰۰ در نظر می‌گیریم). حال اگر شبکه را رسم کنیم و وزن ها را برابر با ۱ در نظر بگیریم، شکل زیر حاصل می‌شود:



$$\mu = e^{-\gamma\|x-v\|^2}$$



به این ترتیب مدل xnor با استفاده از RBF ساخته می‌شود. حال به عنوان مثال شبکه را با ورودی x1 = 1 و x2 = 0 تست می‌کنیم.

$$output(1,0) = w_1 e^{-\gamma||x-v_1||^2} + w_2 e^{-\gamma||x-v_2||^2} = e^{-\gamma\times1} + e^{-\gamma\times1} = 2e^{-100} \cong 0$$

همچنین اگر شبکه را به ازای x1 = 1 و x2 = 1 تست کنیم داریم:

$$output(1,1) = w_1 e^{-\gamma||x-v_1||^2} + w_2 e^{-\gamma||x-v_2||^2} = e^{-\gamma\times0} + e^{-\gamma\times2} = 1 + e^{-200} \cong 1$$

سوال ۵. به سوال‌های زیر درباره‌ی شبکه‌های عصبی پیچشی پاسخ دهید.

الف) Feature extraction را چیست و در کدام مراحل (لایه‌ها) شبکه‌ی عصبی پیچشی انجام می‌شود؟

استخراج ویژگی فرآیندی است که در آن ویژگی های مهم از داده های ورودی مشخص می‌شوند و دیگر نیازی نیست تمام ویژگی ها مورد بررسی قرار گیرند. در شبکه های عصبی پیچشی، مهمترین نوع داده های ورودی، تصاویر هستند و برای تشخیص تصاویر و تمایز میان آنها یک سری ویژگی های خاص را بررسی میکنیم. برای مثال برای تمایز میان گربه و سگ به گوش ها، دهان و ... توجه میکنیم. لذا برای بدست آوردن این ویژگی ها از استخراج ویژگی استفاده میکنیم. فواید استفاده از استخراج ویژگی نیز کاهش ابعاد؛ جدا کردن ویژگی های مختلف، کاهش نویز و ... می‌باشد. در شبکه های عصبی پیچشی در ابتدا داده های ورودی را با استفاده از استخراج ویژگی به داده های مفیدتر و با ابعادی کوچکتر تبدیل می‌کنیم سپس آن را به عنوان ورودی به شبکه عصبی دیگری می‌دهیم.

ب) Max pooling چیست و به چه هدفی انجام می‌شود؟

یکی از مراحل استخراج ویژگی است که در آن با استفاده از یک کرنل با ابعادی مشخص و کوچکتر از داده ورودی بر روی پیکسل های داده ورودی جابجا می‌شود و در هر مرحله تمامی پیکسل هایی که در داخل کرنل قرار میگیرند را با مقدار بزرگترین پیکسل جایگذاری میکنیم. این کار باعث کاهش ابعاد تصاویر با حذف ویژگی های مهم میشود.

پ) ورودی یک شبکه‌ی عصبی پیچشی، یک تصویر رنگی RGB به ابعاد ۹۰۰۹۰۰ است. اگر از ۵۰ فیلتر با ابعاد ۱۰۱۰ استفاده شود، تعداد پارامترهای لایه‌ی پنهان با در نظر گرفتن بایاس چقدر است؟

output = (n*m*l + 1) * k

n = m = 10

l = 3

k = 50

output = (10*10*3 + 1) * 50 = (301) * 50 = 15050

ت) اگر ورودی یک شبکه‌ی عصبی پیچشی تصویری با ابعاد ۱۵۱۵۳ باشد و اندازه‌ی لایه‌گذاری برابر با ۳ باشد، ابعاد تصویر را بعد از گذر از ۴ فیلتر ۳۳ با اندازه گام ۲ به دست بیاورید.

dimension = $\lfloor(n+2p - f)/s + 1\rfloor = \lfloor(n+2p - f)/s + 1\rfloor = \lfloor(15+6-3)/2 + 1\rfloor = 10$

output image dimension = 10 * 10 * 4

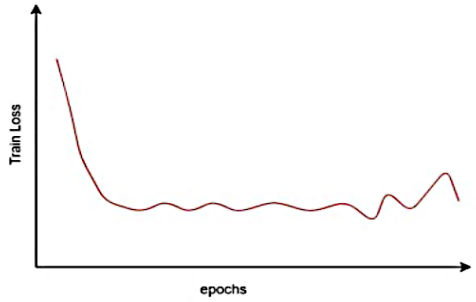
که در آن n بعد ورودی، p اندازه ی لایه گذاری، f اندازه فیلتر و s اندازه گام می‌باشد.

سوال ۶. به سوال‌های زیر درباره‌ی هایپرپارامترها پاسخ دهید.

الف) هایپرپارامتر را توضیح دهید و برای آن چند مثال ذکر کنید.

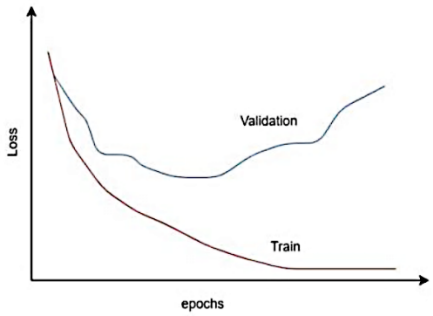
هایپرپارامتر ها در واقع پارامتر های موجود در شبکه های عصبی مانند: وزن لایه ها و بایاس ها را کنترل میکنند. از هایپرپارامتر ها میتوان به نرخ یادگیری، تعداد تکرار ها، تعداد لایه های پنهان و تعداد نورون های آنها، نوع تابع فعال سازی، ضریب ممنتوم، اندازه mini batch و ... اشاره کرد. مقدار بهینه این پارامتر ها باید با تکرار مقادیر مختلف آنها و آزمایش های مختلف تعیین شود.

ب) در نمودار۱ چه مشکلی در هایپرپارامترها وجود دارد؟



مشکلی که وجود دارد این است که هنگام مشتق گرفتن پارامتر ها، مقدار جدید آنها حول نقطه بهینه نوسان میکنند و مهمترین علت آن زیاد بودن نرخ یادگیری است که باید مقدار آن را کاهش دهیم.

پ) پس از بهبود کد، مقدار خطا روی داده‌های آموزشی نزدیک به صفر می‌شود و نمودار۲ حاصل می‌گردد. مشکل این نمودار چیست و چگونه می‌توان آن را برطرف کرد؟



مشکلی که رخ داده است بیش برازش است. یعنی مدل برروی داده های آموزشی خیلی خوب عمل میکند ولی برروی داده های تست دقت خوبی ندارد. برای برطرف کردن مشکل میتوانیم تعداد نورون ها و تعداد لایه هارا کاهش دهیم، داده های آموزشی را بیشتر کنیم، تعدادی از ویژگی هارا حذف کنیم ، از regularization استفاده کنیم و یا...

ت) یکی از مشکلات روش Gradient Descent نوسان حول نقطه‌ی مینیمم است. چگونه می‌توان با استفاده از ضریب یادگیری این مشکل را برطرف کرد؟

با استفاده از ضریب یادگیری می توان اندازه گام را تعیین نمود. معمولا در ابتدای یادگیری می توان از اندازه قدم های بزرگ تری استفاده کرد و با نزدیک شدن به نقطه کمینه، بر اساس یک قاعده مشخص نرخ یادگیری (و در نتیجه اندازه گام) را کاهش داد. به عنوان مثال، در صورتی که با برداشتن چند قدم، مشاهده کنیم که مقدار تابع هزینه کاهش پیدا نکرده است، می توان نرخ یادگیری را کاهش داد.

سوال‌های امتیازی:

سوال ۱. در مورد تابع فعالیت سافت مکس تحقیق کنید و دلایل استفاده از آن در برخی مسائل به جای توابع فعالیت دیگر را توضیح دهید.

softmax یک تابع ریاضی است که بردار اعداد را به بردار احتمالات تبدیل می کند، جایی که احتمالات هر مقدار متناسب با مقیاس نسبی هر مقدار در بردار است. رایج ترین استفاده از تابع softmax در یادگیری ماشین کاربردی، استفاده از آن به عنوان یک تابع فعال سازی در مدل شبکه عصبی است. به طور خاص، شبکه برای خروجی N مقدار، یک عدد برای هر کلاس در کار طبقه بندی، پیکربندی شده است، و تابع softmax برای عادی سازی خروجی ها استفاده می شود، و آنها را از مقادیر جمع وزنی به احتمالات مجموع به یک تبدیل می کند. هر مقدار در خروجی تابع softmax به عنوان احتمال عضویت برای هر کلاس تفسیر می شود.

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \text{ for j = 1,...,K}$$

softmax برای طبقه بندی چند کلاسی استفاده می شود و احتمال هر کلاس را برمیگرداند، در حالی که Sigmoid برای طبقه بندی باینری در مدل رگرسیون لجستیک استفاده می شود.

سوال ۲. به سوال‌های زیر درباره‌ی نرمال‌سازی پاسخ دهید.

الف) نرمال‌سازی در شبکه‌های عصبی چیست و چرا انجام می‌شود؟

یکی از روش های مقیاس کردن داده های یک ویژگی در یک بازه کوچک است که این بازه معمولا [۱, ۱-] یا [۱, ۰] میباشد و با این روش همه داده های در یک بازه مشخص قرار میگیرند و نرمال سازی زمانی کاربرد دارد که دیتاست ما شامل ویژگی هایی با مقیاس های متفاوت باشد و بدون نرمال سازی باعث میشود کارهای مربوط به داده کاوی و ساخت مدل های شبکه های عصبی با دقت خوبی صورت نگیرد.

از روش های نرمال سازی میتوان به decimal scaling, Min-Max, Z-score و ... اشاره کرد.

یکی از دلایل استفاده از نرمال سازی بهبود عملکرد شبکه های عصبی است. برای مثال اگر فرض کنیم از تابع فعال ساز sigmoid و یا tanh استفاده کرده ایم، در صورتی که وزن ها و بایاس ها مقادیر بزرگی داشته باشند باعث میشود مشتق آنها به صفر میل کند و این مورد باعث میشود فرآیند بهینه کردن پارامتر ها به درستی صورت نگیرد و بسیار کند باشد و شبکه عصبی عملکرد خوبی نداشته باشد.

ب) نرمال‌سازی دسته‌ای و دلیل استفاده از آن در شبکه‌های عصبی را توضیح دهید.

یکی از روش های نرمال سازی است که در آن نرمال سازی با استفاده از میانگین و انحراف معیار batch ها انجام میشود و باعث می‌شود فرآیند مشتق گیری با سرعت بهتری انجام شود و عملکرد بسیار خوبی از خود نشان می‌دهد.