یادگیری ژرف

تمرین اول

جواد راضی (۴۰۱۲۰۴۳۵۴)

سوال اول: مرور جبر خطی

۱)

می‌دانیم ماتریس Hessian یک تابع (که تبدیل خطی نیز شاملش می‌شود)، یک ماتریس مربعی شامل تمام مشتقات جزئی مرتبه دوم آن تابع نسبت به متغیرهایش است. برای تبدیل ذکر شده، داریم:

$$H\big(\psi(u,v,z)\big) = \begin{bmatrix} \frac{\partial^2\psi}{\partial u^2} & \cdots & \frac{\partial^2\psi}{\partial u\,\partial z} \\ \vdots & \frac{\partial^2\psi}{\partial v^2} & \vdots \\ \frac{\partial^2\psi}{\partial z\,\partial u} & \cdots & \frac{\partial^2\psi}{\partial z^2} \end{bmatrix}$$

این ماتریس، واضحا یک ماتریس متقارن است.

از طرفی، گرادیان این تبدیل بردار مقابل است:

$$\Delta\psi(u,v,z) = \begin{bmatrix} \frac{\partial\psi}{\partial u} \\ \frac{\partial\psi}{\partial v} \\ \frac{\partial\psi}{\partial z} \end{bmatrix}^T$$

ماتریس ژاکوبیان، که ماتریسی است از مشتقات جزئی مرتبه اول نسبت به متغیرهای یک تابع برداری محاسبه می‌گردد. از آن‌جایی که گرادیان تبدیل، یک تبدیل $R^3 \to R^3$ است، ماتریس ژاکوبیان آن یک ماتریس مربعی ۳ در ۳ خواهد بود .

$$J\big(\Delta\psi(u,v,z)\big) = J\left(\left[\frac{\partial\psi}{\partial u}, \frac{\partial\psi}{\partial v}, \frac{\partial\psi}{\partial z}\right]\right) = \left[\Delta\left(\frac{\partial\psi}{\partial u}\right), \Delta\left(\frac{\partial\psi}{\partial v}\right), \Delta\left(\frac{\partial\psi}{\partial z}\right)\right] = \begin{bmatrix} \frac{\partial^2\psi}{\partial u^2} & \cdots & \frac{\partial^2\psi}{\partial u\,\partial z} \\ \vdots & \frac{\partial^2\psi}{\partial v^2} & \vdots \\ \frac{\partial^2\psi}{\partial z\,\partial u} & \cdots & \frac{\partial^2\psi}{\partial z^2} \end{bmatrix} = H(\psi)$$

بنابراین، نتیجه گرفته می‌شود که ماتریس Hessian یک تبدیل، برابر با ماتریس ژاکوبیان گرادیان آن تبدیل خواهد بود. در خصوص ترتیب مشتقات، از قضیه شوارتز[1] نتیجه گرفته می‌شود که مشتق دوم تابع خاصیت تقارنی دارد و ترتیب مشتق گرفتن مهم نیست. بنابراین، در گرفتن مشتق جزئی از بردار گرادیان تبدیل، می‌توان ترتیب منطبق را ماتریس Hessian را در نظر گرفت.

۲)

الف)

$$x^T a = a^T x = \sum_1^n x_i a_i \rightarrow$$

$$\frac{\partial x^T a}{\partial x} = \frac{\partial a^T x}{\partial x} = \left[\frac{\partial \sum_1^n x_i a_i}{\partial x_1}, \dots, \frac{\partial \sum_1^n x_i a_i}{\partial x_n}\right] = [a_1, a_2, \dots, a_n] = a$$

ب)

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

$$tr\left(\frac{\partial X}{\partial y}\right) = tr\left(\begin{pmatrix} \frac{\partial x_{11}}{\partial y} & \cdots & \frac{\partial x_{1n}}{\partial y} \\ \vdots & \ddots & \vdots \\ \frac{\partial x_{n1}}{\partial y} & \cdots & \frac{\partial x_{nn}}{\partial y} \end{pmatrix}\right) = \sum_1^n \frac{\partial x_{kk}}{\partial y} = \frac{\partial}{\partial y} \sum_1^n x_{kk} = \frac{\partial}{\partial y} tr(X)$$

پ)

می‌دانیم trace، خاصیت تقارنی برای ضرب دارد. پس داریم:

$$tr(X^T A X) = tr(A X X^T)$$

$$(AXX^T)_{ij} = A_i \,.\, (XX^T)_j = A_i \,.\, \begin{bmatrix} X_1 \,.\, X_j \\ X_2 \,.X_j \\ \dots \\ X_n \,.\, X_j \end{bmatrix} = \sum_{k=1}^n A_{ik} \sum_{l=1}^n X_{kl} X_{jl}$$

$$\rightarrow\ tr(AXX^T) = \sum_{i=1}^n \sum_{k=1}^n \sum_{l=1}^n A_{ik} X_{kl} X_{il} \rightarrow$$

[1]

$$\frac{\partial tr(AXX^T)}{\partial X_{pq}} = \frac{\partial(\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} A_{ik}X_{kl}X_{il})}{\partial X_{pq}} = X_{pq}\left(A_{pq} + A_{pq}\right) = [X^T(A + A^T)]_{pq}$$

ت)

مطابق خواص مشتق:

$$\frac{\partial \log(\det(X))}{\partial X} = \frac{1}{\det(X)} \cdot \frac{\partial}{\partial X}\det(X)$$
$$\rightarrow (using\ Jacobi's\ Formula\ for\ derivative\ of\ Determinant) \rightarrow$$
$$= \frac{1}{\det(X)} \cdot \left(adj(X)\right)^T = \frac{1}{\det(X)}(C^T)^T = (X^{-1})^T = X^{-T}$$

۳)

برای یافتن مقادیر ویژه، معادله مشخصه ماتریس را حل می‌کنیم:

$$\det(R(\theta) - \lambda I) = 0 \rightarrow \begin{vmatrix} \cos(\theta) - \lambda & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) - \lambda \end{vmatrix} = 0 \rightarrow$$
$$(\cos(\theta) - \lambda)^2 + \sin^2(\theta) = 0 \rightarrow \lambda^2 - 2\lambda\cos(\theta) + 1 = 0 \rightarrow$$
$$\begin{cases} \lambda_1 = \cos(\theta) + isin(\theta) \\ \lambda_2 = \cos(\theta) - isin(\theta) \end{cases}$$

برای یافتن بردارهای ویژه، معادله زیر را حل می‌کنیم:

$$(R(\theta) - \lambda_1 I)v = 0 \rightarrow$$
$$\begin{bmatrix} -isin(\theta) & -\sin(\theta) \\ \sin(\theta) & -isin(\theta) \end{bmatrix} \cdot \begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = 0 \rightarrow \quad v_1 = \begin{bmatrix} 1 \\ -i \end{bmatrix}$$

می‌دانیم اگر مقادیر ویژه یک ماتریس مزدوج مختلط باشند، بردارهای ویژه آن نیز چنین خواهند بود.
بنابراین، مقدار بردار ویژه دیگر برابرست با:

$$v_2 = \begin{bmatrix} 1 \\ i \end{bmatrix}$$

برای این ماتریس، دترمینان، از رابطه زیر به‌دست می‌آید:

$$\det\left(R(\theta)\right) = \cos^2(\theta) + \sin^2(\theta) = 1$$

$$\lambda_1 * \lambda_2 = e^{i\theta}.e^{(-i\theta)} = 1 = \det\big(R(\theta)\big)$$

قسمت آخر: قطری‌سازی ماتریس:

$$R(\theta) = PD(\theta)P^{-1} \rightarrow R^n(\theta) = PD(\theta)^n P^{-1} = P\begin{bmatrix} e^{i\theta} & \\ & e^{-i\theta} \end{bmatrix}^n P^{-1} = P\begin{bmatrix} e^{in\theta} & \\ & e^{-in\theta} \end{bmatrix} P^{-1} = R(n\theta)$$

سوال دوم: بهینه‌سازی

۱)

به صورت شهودی، هم برای نقطه زینی، و هم برای کمینه محلی، شرط صفر بودن مشتق جزئی در تمام جهات وجود دارد. اما برای کمینه محلی بودن، باید ماتریس Hessian، مثبت معین باشد، یعنی تمام مقادیر ویژه آن باید مثبت باشند. [2] این در شرایطی است که برای نقطه زینی‌بودن، باید ماتریس Hessian هم مقادیر ویژه مثبت، و هم مقادیر ویژه منفی داشته باشد. از دید احتمالی، هرچه تعداد ابعاد بالاتر می‌رود، احتمال اینکه تمام مقادیر ویژه ماتریس هم‌علامت نباشند، بالاتر می‌رود. در واقع، با بالاتر رفتن ابعاد، احتمال اینکه با حرکت در حداقل یک جهت، علامت مشتق خلاف جهات دیگر شود، بالاتر از اینکه علامت مشتق در حرکت در تمام جهات یکسان باشد خواهد بود.

۲)

الف)

نمودار مشکی، که تقریبا با مسیری مستقیم به نقطه بهینه رسیده‌است، احتمالا مربوط به روش RMSprop است. این روش، که یک روش تطبیقی برای نرمال‌سازی نرخ یادگیری با میانگین متحرک (Moving Average) مربع گرادیان‌هاست، نرخ یادگیری هر وزن را، بر اساس این میانگین نمایی،

---

[2] https://en.wikipedia.org/wiki/Second_partial_derivative_test#Functions_of_many_variables

اسکیل می‌کند. در چنین روشی، و برای تابع مرتبه دو، با هایپرپارامترهای معقول، احتمال اینکه مسیرمان به نقطه بهینه مستقیم باشد، بیشتر از روش‌های دیگر است.

نمودار سبز، احتمالا مربوط به روش momentum است. در این روش، با معرفی مفهوم تکانش و velocity، گرادیان‌های قبلی در آپدیت velocity دخیل هستند. بنابراین، هر چه در مسیری مستمر حرکت کنیم، وقتی به نقطه‌ای برسیم که گرادیان محاسبه‌شده، جهت را تغییر می‌دهد، به خاطر velocity و تاثیر گرادیان‌های قبل، محتمل است که چند گام دیگر در راستای مسیر پیشین برداریم، پیش از آنکه مسیر تغییر کند. این تفسیر از نحوه کارکرد روش momentum در این مورد خاص، با نمودار سبز همخوانی دارد.

نمودار قرمز، احتمالا مربوط به روش Nestrov-Momentum است. این روش، عملکرد روش momentum را اندکی بهبود می‌بخشد و با دخالت‌دادن velocity در محاسبه گرادیان، جهت حرکت کنونی را اندکی تغییر می‌دهد. این امر، منجر به بهبود در سرعت همگرا شدن، و کاهش زیگزاگ می‌شود. نمودار قرمز، با نحوه عملکرد این روش هم‌خوان است.

ب)

مشکل روش GD، کند بودن آن، و احتمال گیر کردن در نقطه مینیمم محلی است. در خصوص نحوه کارکرد هر ۳ روش و مزایا و معایب آن‌ها، در قسمت قبل تا حدی توضیح داده شد. به طور خلاصه، در مقایسه با GD، هر یک از روش‌ها این مزایا و معایب را دارند:

- روش Momentum، با دخالت‌دادن گرادیان‌های قبلی، در حرکت یک momentum ایجاد می‌کند که منجر می‌شود از نقاط مینیمم محلی سطحی، به راحتی عبور کرد. با این حال، این momentum، همان‌گونه که در مثال تصویری دیدیم، ممکن است سبب شود که تغییر در جهت حرکت، به علت جمع‌شدن تاثیرات گرادیان‌های قبلی، کندتر انجام شود.
- روش Nestrov-Momentum، مزیت روش Momentum را دارد و عملکرد آن‌را تا حدی بهبود می‌بخشد. این روش، با دخالت velocity کنونی در محاسبه گرادیان، به نوعی با آینده‌نگری، مسیر همگرایی را Smoothتر، و همگرایی را سریع‌تر می‌کند. با این حال، این روش هم می‌تواند زیگزاگ‌های بی‌هدف را در پی داشته باشد که با وجود همگرا شدن، مسیر همگرا شدن را غیربهینه می‌کند.
- روش RMSprop، با Adapt کردن نرخ یادگیری برای وزن‌های مختلف، منجر به نوسان کمتری شده و سرعت همگرایی را نیز بهبود می‌بخشد. در عین حال، این روش می‌تواند به انتخاب

هایپرپارامترهایی نظیر نرخ میرایی (در محاسبه میانگین نمایی)، و نرخ یادگیری اولیه، بیشتر از روش‌های قبل حساس باشد.

پ)

روش ADAM، به نوعی دو روش Momentum، و RMSprop را ترکیب کرده و ضعف‌های روش Momentum را بهبود می‌بخشد؛ یک مشکل اساسی روش Momentum این است که ممکن است حتی موقع رسیدن به نقطه مینیما، به علت Momentumیی که جمع‌شده، از آن نقطه بگذرد. در روش ADAM، علاوه بر Momentum (موسوم به moment اول)، میانگین نمایی مربع گرادیان‌ها در روش RMSprop نیز معرفی شده‌است. مطابق روش Momentum، گرادیان‌های قبلی نیز با میانگین نمایی در بروزرسانی وزن‌ها دخیل هستند. اما میزان بروزرسانی، توسط جذر ترم دوم، نرمال‌سازی می‌شود تا از Overshoot کردن نقطه مینیما جلوگیری شود.

در ابتدای یادگیری، بردارهای میانگین نمایی در روش ADAM، با مقادیر صفر راه‌اندازی می‌شوند. این باعث می‌شود که در محاسبه میانگین، بایاس به سمت صفر داشته باشیم چرا که در این محاسبه آخرین گرادیان در یک ترم decay ضرب می‌شود. روش ADAM، با معرفی دو پارامتر $\mathrm{B}_1, \mathrm{B}_2$، که با توان t که نمایان‌گر زمانی که از ترین‌شدن گذشته‌است آپدیت می‌شوند، مقادیر moment اول و moment دوم را آپدیت می‌کند. (مثال: $\hat{m} = \frac{m}{1-\mathrm{B}_1^t}$) در استیج‌های آغازین ترین‌شدن، بایاس به سمت صفر این مقادیر، توسط دو پارامتر B_1, B_2 تصحیح می‌گردد. W

سوال سوم: منظم‌سازی

۱)

متدهای Ensemble، با تجمیع و اثردادن به خروجی چند مدل که در ساختار و معماری مدل، یا داده ورودی تفاوت داشتند، تلاش می‌کردند واریانس را کاهش داده و از Overfit شدن مدل جلوگیری نمایند. مطابق مقاله معرفی‌شده، تکنیک Dropout، که در آن نورون‌هایی، به همراه تمام کانکشن‌هایشان به صورت رندم در حین آموزش دراپ می‌شوند، معادل یادگیری تعداد نمایی از مدل‌های با وزن مشابه، و میانگین‌گرفتن از خروجی آن‌ها عمل می‌کند، که کاری معادل روش‌های Ensemble است.

علت برتری این متد به روش‌های Ensemble در شبکه‌های با تعداد پارامتر بالا، اینست که این متد، از لحاظ محاسباتی بسیار Efficientتر است. استفاده از روش‌های Ensemble، نیاز به آموزش و نگه‌داری وزن‌های تعداد بالایی مدل شبکه عصبی عمیق، با تعداد بسیار بالای پارامتر دارد. تکنیک Dropout، عملا همین کار را شبیه‌سازی می‌کند، و در واقع از نتیجه تعداد بالایی از شبکه‌های عصبی سبک‌تر شده،

میانگین می‌گیرد. اما این‌کار، در حین یک بار آموزش شبکه عصبی اتفاق می‌افتد که بسیار کاراتر، و پرکتیکال است.

۲)

با افزودن نویز و Randomness به فرایند یادگیری مدل، تکنیک Dropout از Overfit شدن مدل جلوگیری می‌کند؛ به بیان دیگر، با حذف رندم تعدادی نورون در حین آموزش، شبکه یاد می‌گیرد که با خروجی یک نورون وابسته نشود، و بر اساس خروجی یک نمونه از نورون‌های ورودی، فیچرهای مستقلی از ورودی را یاد گیرد. این عمل، که به کاهش معنادار احتمال Overfit شدن مدل منجر می‌شود، تکنیک Dropout را معادل یک تکنیک منظم‌سازی می‌کند.

۳)

تفاوت اساسی استفاده از تکنیک Dropout در ترین و تست مدل، اینست که در هنگام تست، Dropout اعمال نمی‌شود و هیچ واحدی از شبکه به صورت رندم حذف نمی‌گردد؛ چرا که می‌خواهیم از تمام قدرت شبکه برای پیش‌بینی استفاده کنیم. با این حال، به علت اینکه در حین ترین‌شدن نورون‌هایی به همراه کانکشن‌هایشان با احتمال معینی دراپ شده‌اند، در هنگام تست، وزن‌ها، با نسبت احتمال Dropout، Scale-Down می‌شوند تا مقدار مورد انتظار خروجی هر واحد، در ترین و تست برابر باشد.

۴)

۵)

- با نرمال‌سازی توزیع ورودی هر لایه، تکنیک Batch-Normalization، می‌تواند ریسک تغییر داخلی Covariance (Internal Covariance Shift)، که می‌تواند به خاطر تغییر در توزیع داده‌های ورودی

اتفاق بیفتد را کم می‌کند.[3] به همین خاطر، می‌تواند از مقادیر بالاتری برای نرخ یادگیری، بدون نگرانی از خطر واگرایی استفاده نمود.

- این تکنیک، با نگه‌داشتن متوسط، و واریانس دیتای ورودی در یک رنج خاص، باعث پایداری مدل می‌شود، و احتمال مشکلاتی نظیر Vanishing Gradient، و Exploding Gradient را کم می‌کند. کاهش ریسک این خطرات، به ما آزادی بیشتری در کنترل نرخ یادگیری بدون نگرانی در خصوص این موارد می‌دهد.

۶)

تکنیک Batch-Normalization، با محاسبه میانگین و واریانس هر mini-batch، و نرمال‌سازی داده‌های mini-batch بر اساس این مقادیر، به گونه‌ای نویز وارد مدل می‌کند. چرا که هر بچ، میانگین و واریانس خود را خواهد داشت، و داده‌های آن بر اساس این میانگین و واریانس نرمال‌سازی خواهند شد. (نه میانگین و واریانس کل نمونه‌ها). این تفاوت در پارامترهای نرمال‌سازی هر بچ را می‌توان به افزودن نویز، و رندم‌نس به دیتا تفسیر کرد، که می‌تواند که لایه‌های مدل را وادار به این کند که فیچرهای اصلی‌تر، و منسجم‌تری را یاد بگیرند و Generalization بهتر انجام شود.

با بزرگ‌تر شدن سایز بچ، میانگین و واریانس تخمینی، به میانگین و واریانس واقعی نمونه‌ها نزدیک‌تر شده و نرمال‌سازی، دارای نویز و رندم‌نس پایین‌تری خواهد بود. به عنوان مثال اگر ۱۰۰۰ نمونه داشته‌باشیم و سایز بچ ۵۰۰ باشد، بسیار محتمل است که میانگین و واریانس هر بچ بسیار نزدیک به هم، و نزدیک به میانگین و واریانس کل نمونه‌ها باشد. (نسبت به بچ ۵۰تایی). این امر با کاهش اثر نویز و رندم‌نس، اثر منظم‌سازی Batch-Normalization را کم‌تر می‌کند، و احتمال Overfitشدن مدل بالاتر می‌رود. در عین حال و از سوی دیگر، ترین‌شدن مدل پایداری بیشتری داشته و احتمال همگرایی بیشتر می‌شود، چرا که بچ‌ها توزیع مشابه یکدیگر دارند و این امر پایداری را بالاتر می‌برد.

[3] https://machinelearning.wtf/terms/internal-covariate-shift/

سوال چهارم: توابع فعال‌سازی

۱)

آ)

تابع سیگموید برای این طبقه‌بندی باینری مناسب است. خروجی تابع سیگموید یک عدد بین ۰ تا ۱ است که یک مقدار احتمالی را نشان می‌دهد. برای تسک دسته‌بندی باینری بین سگ و گربه، این تابع بهینه است.

ب) تابع Softmax برای این طبقه‌بندی چندکلاسه مناسب‌تر است. خروجی این تابع، یک توزیع احتمالاتی، با ابعاد به اندازه تعداد کلاس‌هاست که برای هر کلاس، یک احتمال به آن نسبت می‌دهد و مجموع تمام احتمال‌ها، یک خواهد بود.

ج)

یک تابع سیگموید، به ازای هر نورون در لایه خروجی (به تعداد کلاس‌های موجود) برای این مسئله مناسب است. از آن‌جایی که مسئله طبقه‌بندی چندکلاسه و چندلیبله است، هر نورون با تابع سیگموید، به عنوان نماینده یک کلاس عمل می‌کند که می‌تواند مقداری بین صفر و یک را داشته‌باشد و بیانگر حضور، یا عدم حضور یک حیوان در عکس باشد.

۲)

تابع ReLU، یکی از پرکاربردترین توابع در لایه‌های پنهان شبکه‌های عصبی عمیق است؛ اما صفر بودن مقدار این تابع برای مقادیر کوچک‌تر از صفر، می‌تواند مشکل‌ساز شود. (موسوم به مشکل Dying ReLU). برای حل این مسئله، Variantهای مختلفی از این تابع نظیر LeakyReLU معرفی شده‌اند. تابع DPReLU (Dynamic Parameter ReLU)، به صورت پویا، در طی آموزش، شکل خود را تغییر می‌دهد. این تابع، ۴ پارامتر قابل یادگیری در طی آموزش را اضافه می‌کند. با اضافه‌کردن این پارامترها، بسته به مدل و دامنه مسئله، می‌توان نسخه‌ای منعطف از ReLU، که برای مدل بهینه است را یاد گرفت. چهار پارامتر قابل یادگیری این تابع عبارتند از:

- آلفا، که شیب قسمت منفی تابع را کنترل می‌کند.
- بتا، که شیب قسمت مثبت تابع را کنترل می‌کند.
- بایاس، که تابع را به بالا یا پایین شیفت می‌دهد.

- آستانه (Threshold)، که تابع را به چپ یا راست شیف می‌دهد و در واقع آستانه‌ای که مرز میان قسمت مثبت و منفی تابع را کنترل می‌کند را تعیین می‌کند.

نسخه‌های قبلی این تابع، از جمله PReLU و FReLU، تنها دو تا از چهار پارامتر قابل یادگیری (آلفا، و بایاس) داشتند. این موضوع، منجر به این شده‌بود که تابع انعطاف‌پذیری و تطبق‌پذیری کم‌تری داشته‌باشد. همچنین، شیب قسمت مثبت تابع نیز نادیده گرفته‌شده بود. از سوی دیگر، در این مقاله، روشی جدید برای مقداردهی اولیه به وزن‌ها معرفی شده تا جلوی اشباع تابع فعال‌سازی گرفته شود؛ در این روش، وزن‌های توابع بر اساس خروجی مورد انتظار آن‌ها مقداردهی می‌گردند تا از اشباع‌شدن توابع جلوگیری شود و سرعت همگرایی مدل افزایش یابد.

سوال پنجم: شبکه‌های عصبی و انتشار رو به عقب

۱)

مرحله یک: محاسبه Forward Propagation:

$$z_{h1} = w_1 \cdot x_1 + w_3 \cdot x_2 + b_1 = 0.3 \cdot 0 + 0.2 \cdot 1 + 0.2 = 0.4$$
$$z_{h2} = w_2 \cdot x_1 + w_4 \cdot x_2 + b_2 = 0.2 \cdot 0 - 0.6 \cdot 1 - 1.4 = -2$$

$$h_1 = LreLU(z_{h1}) = 0.4 \; (z_{h1} \geq 0)$$
$$h_2 = LreLU(z_{h2}) = 0.2 \cdot -2 = -0.4 \;\; (z_{h2} < 0)$$

$$z_y = w_5 \cdot h_1 + w_6 \cdot h_2 = 0.5 \cdot 0.4 - 1 \cdot -0.4 = 0.6$$

$$\hat{y} = sigmoid(z_y) = \frac{1}{1 + e^{-0.6}} \approx 0.65$$

$$L = \frac{1}{2}(\hat{y} - y)^2 = \frac{1}{2}(0.65 - 1)^2 \approx 0.06$$

مرحله دو: Backward Propagation:

$$\frac{dL}{d\hat{y}} = \hat{y} - y = 0.65 - 1 = -0.35$$
$$\frac{d\hat{y}}{dz_y} = \hat{y} \cdot (1 - \hat{y}) = 0.65 \cdot (1 - 0.65) \approx 0.23$$
$$\frac{dz_y}{dw_5} = h_1 = 0.4$$
$$\frac{dz_y}{dw_6} = h_2 = -0.4$$
$$\frac{dz_y}{dh_1} = w_5 = 0.5$$
$$\frac{dz_y}{dh_2} = w_6 = -1$$

$$\frac{dh_1}{dz_{h1}} = 1\ (z_{h1} \geq 0)$$
$$\frac{dh_2}{dz_{h2}} = 0.2\ (z_{h2} < 0)$$

$$\frac{dz_{h1}}{dw_1} = x_1 = 0$$
$$\frac{dz_{h1}}{dw_3} = x_2 = 1$$
$$\frac{dz_{h1}}{db_1} = 1$$
$$\frac{dz_{h2}}{dw_2} = x_1 = 0$$
$$\frac{dz_{h2}}{dw_4} = x_2 = 1$$
$$\frac{dz_{h2}}{db_2} = 1$$

مرحله سه: آپدیت پارامترها:

$$w_5 = w_5 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dw_5} \approx 0.5 - 0.1 \cdot -0.35 \cdot 0.23 \cdot 0.4 \approx 0.51$$

$$w_6 = w_6 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dw_6} \approx -1 - 0.1 \cdot -0.35 \cdot 0.23 \cdot -0.4 \approx -1.01$$

$$w_1 = w_1 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_1} \cdot \frac{dh_1}{dz_{h1}} \cdot \frac{dz_{h1}}{dw_1} = 0.3 \text{ (since } x_1 = 0)$$

$$w_2 = w_2 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_2} \cdot \frac{dh_2}{dz_{h2}} \cdot \frac{dz_{h2}}{dw_2} = 0.2 \text{ (since } x_1 = 0)$$

$$w_3 = w_3 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_1} \cdot \frac{dh_1}{dz_{h1}} \cdot \frac{dz_{h1}}{dw_3} \approx 0.2 - 0.1 \cdot -0.35 \cdot 0.23 \cdot 0.5 \cdot 1 \cdot 1 \approx 0.21$$

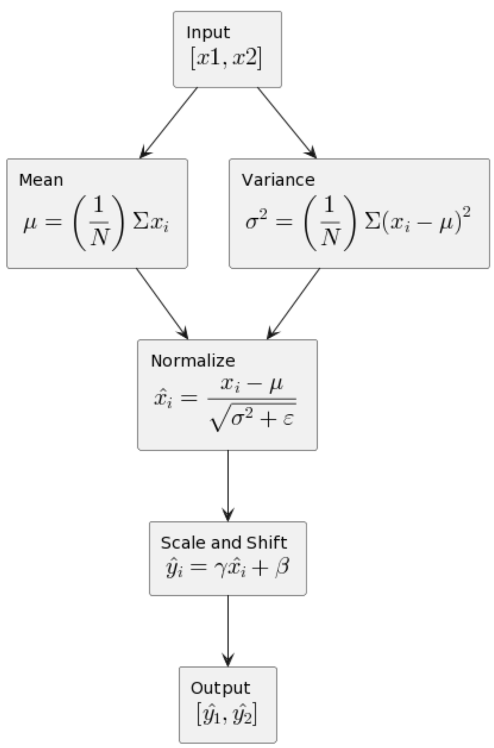
$$w_4 = w_4 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_2} \cdot \frac{dh_2}{dz_{h2}} \cdot \frac{dz_{h2}}{dw_4} \approx -0.6 - 0.1 \cdot -0.35 \cdot 0.23 \cdot -1 \cdot 0.2 \cdot 1 \approx -0.59$$

$$b_1 = b_1 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_1} \cdot \frac{dh_1}{dz_{h1}} \cdot \frac{dz_{h1}}{db_1} \approx 0.2 - 0.1 \cdot -0.35 \cdot 0.23 \cdot 0.5 \cdot 1 \cdot 1 \approx 0.21$$

$$b_2 = b_2 - \alpha \cdot \frac{dL}{d\hat{y}} \cdot \frac{d\hat{y}}{dz_y} \cdot \frac{dz_y}{dh_2} \cdot \frac{dh_2}{dz_{h2}} \cdot \frac{dz_{h2}}{db_2} \approx -1.4 - 0.1 \cdot -0.35 \cdot 0.23 \cdot -1 \cdot 0.2 \cdot 1 \approx -1.39$$

۲)

گراف محاسباتی:



برای هر گره گراف محاسباتی داریم:

$$\mu = \left(\frac{1}{N}\right)\sum x_i$$

$$\sigma^2 = \left(\frac{1}{N}\right)\sum (x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$\hat{y}_i = \gamma \hat{x}_i + \beta$$

محاسبه روابط مشتقات هزینه‌ها:

$$\frac{\partial L}{\partial \beta} = \frac{\partial L}{\partial y_1} + \frac{\partial L}{\partial y_2}$$

$$\frac{\partial L}{\partial \gamma} = \frac{\partial L}{\partial y_1}\hat{x}_1 + \frac{\partial L}{\partial y_2}\hat{x}_2$$

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial \widehat{x_\imath}} \cdot \frac{1}{\sqrt{\sigma^2 + \epsilon}} + \frac{\partial L}{\partial \sigma^2} \cdot \frac{2(x_i - \mu)}{N} + \frac{\partial L}{\partial \mu} \cdot \frac{1}{N}$$

نحوه محاسبه مشتقات داده‌نشده:

$$\frac{\partial L}{\partial \widehat{x_\imath}} = \frac{\partial L}{\partial y_k} \cdot$$

$$\frac{\partial L}{\partial \sigma^2} = \sum_{k=1}^{N} \frac{\partial L}{\partial \widehat{x_k}} \cdot (x_k - \mu) \cdot \frac{-1}{2}(\sigma^2 + \epsilon)^{-3/2}$$

$$\frac{\partial L}{\partial \mu} = \sum_{k=1}^{N} \frac{\partial L}{\partial \widehat{x_k}} \cdot \frac{-1}{\sqrt{\sigma^2 + \epsilon}} + \frac{\partial L}{\partial \sigma^2} \cdot \frac{-2}{N} \sum_{k=1}^{N} (x_k - \mu)$$