دانشگاه صنعتی اصفهان

دانشکده مهندسی برق و کامپیوتر

عنوان: تکلیف دوم درس یادگیری عمیق

نام و نام خانوادگی: علیرضا ابره فروش

شماره دانشجویی: ۹۸۱۶۶۰۳

نیم‌سال تحصیلی: پاییز ۱۴۰۲

مدرّس: دکتر سمانه حسینی سمنانی

دستیاران آموزشی: مریم محمدی-علی بزرگ زادارباب

۱

۲

فرض کنیم که $\hat{y}_i$ها خروجی‌های شبکه باشند. بدون از دست رفتن کلیت فرض می‌کنیم $y_{k_1}, \cdots, y_{k_L}$ برابر یک باشند ($k_1, \cdots, k_L$ کلاس‌های درست باشند). با توجه به اینکه مجموع $\hat{y}_i$ها برابر ۱ است و دقیقا Lتا از y_iها برابر ۱ هستند داریم:

$$L\left(\hat{y}, y\right) = -\sum_{i=1}^{n_y} y_i \log \hat{y}_i = -\sum_{j=1}^{L} y_{k_j} \log \hat{y}_{k_j} = -\sum_{j=1}^{L} \log \hat{y}_{k_j}$$

طبق نامساوی حسابی-هندسی داریم:

$$\frac{\hat{y}_{k_1}+\cdots+\hat{y}_{k_L}}{L} \geq \sqrt[L]{\hat{y}_{k_1}\cdots\hat{y}_{k_L}}$$

$$\xrightarrow{\text{both sides to the power of } L}$$

$$\left(\frac{\hat{y}_{k_1}+\cdots+\hat{y}_{k_L}}{L}\right)^L \geq \hat{y}_{k_1}\cdots\hat{y}_{k_L}$$

$$\xrightarrow{1\geq\hat{y}_{k_1}+\cdots+\hat{y}_{k_L}}$$

$$\left(\frac{1}{L}\right)^L \geq \left(\frac{\hat{y}_{k_1}+\cdots+\hat{y}_{k_L}}{L}\right)^L \geq \hat{y}_{k_1}\cdots\hat{y}_{k_L}$$

$$\xrightarrow{\text{log of each side}}$$

$$-L\log L \geq \log\left(\hat{y}_{k_1}\cdots\hat{y}_{k_L}\right) \Longrightarrow -\log\left(\hat{y}_{k_1}\cdots\hat{y}_{k_L}\right) \geq L\log L$$

$$\Longrightarrow -\sum_{j=1}^{L} \log \hat{y}_{k_j} \geq L\log L$$

$$\Longrightarrow L\left(\hat{y}, y\right) \geq L\log L$$

$\square$

۳

با توجه به اینکه در انتهای نمودار Loss در training بسیار کمتر از validation است، پس شبکه دچار overfitting شده است.

۱.۳

افزایش عمق شبکه عصبی می‌تواند به افزایش overfitting منجر شود. شبکه‌های عمیق‌تر ظرفیت بالاتری برای یادگیری الگوهای پیچیده دارند، اما اگر مجموعه داده کافی بزرگ نباشد، overfitting را تشدید خواهد کرد. اگر overfitting ناشی از ظرفیت شبکه برای یادگیری نویز در داده‌های آموزش باشد، افزایش عمق ممکن است وضعیت را بدتر کند. با این حال، لازم به ذکر است اگر overfitting به دلیل عدم یادگیری مدل از الگوهای موجود باشد، یک شبکه عمیق‌تر ممکن است به تعمیم بیشتر کمک کند.

۲.۳

توقف زودهنگام یک تکنیک جهت جلوگیری از overfitting است که شامل نظارت بر عملکرد مدل در یک مجموعه اعتبارسنجی و متوقف کردن فرآیند آموزش هنگامی که عملکرد شبکه بهبود چشمگیری نمی‌یابد یا شروع به افت می‌کند. این ممکن است در جلوگیری از overfitting مؤثر باشد چرا که به مدل کمک می‌کند که خوب با داده‌های ناشناخته تعمیم پیدا کند. اگر یک مدل در حال overfitting باشد، تکنیک توقف زودهنگام می‌تواند جلوی ادامه یادگیری نویز در داده‌های آموزش را بگیرد و بنابراین توانایی بهتری در تعمیم به داده‌های جدید برای آن ایجاد کند.

۳.۳

آموزش بیشتر بدون هیچ گونه regularization احتمالا overfitting را تشدید می‌کند. ممکن است مدل به طور زیادی به داده‌های آموزش فیت شود و نتواند به داده‌های جدید و ناشناخته تعمیم پیدا کند. با این حال، اگر overfitting ناشی از عدم یادگیری مدل از الگوهای موجود در داده‌ها باشد، آموزش بیشتر ممکن است به مدل کمک کند تا به یک وضعیت بهتر همگرا شود.

۴.۳

Data Augmentation شامل اعمال تبدیلات مختلف بر داده‌های آموزش، مانند چرخش‌ها، وارونه‌ها و جابجایی‌ها می‌شود. این می‌تواند به مدل کمک کند تا مقاومت بیشتری در برابر overfitting پیدا کرده و با تغییرات در داده ورودی بهتر تعمیم پیدا کند. Data Augmentation خصوصاً زمانی که مجموعه داده محدود است، می‌تواند با ارائه مدل به یک طیف گسترده‌تر از تغییرات در داده‌های آموزش، از overfitting کاسته و توانایی بهتری در تعمیم ایجاد کند.

۴

ابتدا تعریف می‌کنیم:

$$\forall i \in \{1, 2, \cdots, K\}: \quad u_i := z_i^{[2]}$$

$$A := \sum_{j=1}^{K} e^{u_j}$$

۱.۴ الف

$$\frac{\partial \hat{y}_k}{\partial z_k^{[2]}} = \frac{\partial \hat{y}_k}{\partial u_k} = \frac{\partial\left(\frac{e^{u_k}}{A}\right)}{\partial u_k} = \frac{\frac{\partial e^{u_k}}{\partial u_k} \times A - \frac{\partial A}{\partial u_k} \times e^{u_k}}{A^2} = \frac{e^{u_k}}{A} - \left(\frac{e^{u_k}}{A}\right)^2 = \hat{y}_k - \hat{y}_k^2$$

۲.۴ ب

$$\frac{\partial \hat{y}_k}{\partial z_i^{[2]}} = \frac{\partial \hat{y}_k}{\partial u_i} = \frac{\partial\left(\frac{e^{u_k}}{A}\right)}{\partial u_i} = \frac{\frac{\partial e^{u_k}}{\partial u_i} \times A - \frac{\partial A}{\partial u_i} \times e^{u_k}}{A^2} = 0 - \left(\frac{e^{u_i} e^{u_k}}{A^2}\right) = -\hat{y}_i \hat{y}_k$$

۳.۴ ج

$$\frac{\partial L}{\partial z_i^{[2]}} = \frac{\partial L}{\partial u_i} = \frac{\partial\left(-\sum_{j=1}^{K} y_j \log \hat{y}_j\right)}{\partial u_i} = -\sum_{j=1}^{K} y_j \times \frac{\partial(\log \hat{y}_j)}{\partial u_i} = -1 \times \frac{\partial(\log \hat{y}_k)}{\partial u_i} = -\frac{\partial(\log \hat{y}_k)}{\partial \hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial u_i} = -\frac{1}{\hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial u_i}$$

$$= \begin{cases} -\frac{1}{\hat{y}_k}\left(\hat{y}_k - \hat{y}_k^2\right) & i = k \\ -\frac{1}{\hat{y}_k}\left(-\hat{y}_i \hat{y}_k\right) & i \neq k \end{cases} = \begin{cases} \hat{y}_k - 1 & i = k \\ \hat{y}_i & i \neq k \end{cases}$$

۴.۴ د

۵.۴ ه

مشکل این تابع در محاسبات عددی از آنجا آغاز می‌شود که اگر z_iها اعداد بزرگی باشند، ممکن است شبکه با مشکلات پایداری عددی روبرو شود. از آنجایی که اعداد ممیز شناور محدودیت دارند، عملیات اعداد ممیز شناور با دقت محدودی انجام شود که ممکن است به کاهش دقت محاسبات منجر شود. این پدیده به عنوان "انفجار گرادیان" نیز شناخته می‌شود.

برای جلوگیری از overflow باید مانع از زیاد بزرگ شدن اعداد شویم. البته underflow هم در این حالت ممکن است رخ دهد و این ایده‌آل ما نیست؛ در این حالت مقادیر عبارات ۰ می‌شوند. اما این بهتر از مقدار بی‌نهایت یا تعریف نشده است.

فرمول اصلاح شده به دلایل زیر می‌تواند به حل این مشکل کمک کند:

۱. از آنجایی که $z_i^{[2]} - m \leq 0$ همه مقادیر $e^{z_i^{[2]}-m}$ بین صفر و یک خواهند بود. این مشکل overflow را برطرف می‌کند (البته مشکل underflow سر جایش باقی است).

۲. حداقل یکی از مقادیر نمایی برابر ۱ است (در حالتی که $z_i^{[2]} = m$ باشد). در این صورت این را تضمین می‌کند که حداقل یک مقدار دچار underflow نمی‌شود. همچنین مخرج کسر همواره بزرگتر مساوی ۱ است که از تقسیم بر صفر جلوگیری می‌کند. و در نهایت حداقل ۱ مقدار غیر صفر در صورت کسر وجود دارد و softmax نمی‌تواند یک بردار صفر خروجی دهد.

۵

۱.۵ الف

$$z = \begin{bmatrix} 12 & 14 & 14 & 12 \\ 0 & 10 & 10 & 0 \\ -5 & 5 & 5 & -5 \end{bmatrix}$$
$$\mu_1 = \frac{12+14+14+12}{4} = 13$$
$$\mu_2 = \frac{0+10+10+0}{4} = 5$$
$$\mu_3 = \frac{-5+5+5-5}{4} = 0$$
$$\sigma_1 = \sqrt{\frac{(12-13)^2+(14-13)^2+(14-13)^2+(12-13)^2}{4}} = 1$$
$$\sigma_2 = \sqrt{\frac{(0-5)^2+(10-5)^2+(10-5)^2+(0-5)^2}{4}} = 5$$
$$\sigma_3 = \sqrt{\frac{(-5-0)^2+(5-0)^2+(5-0)^2+(-5-0)^2}{4}} = 5$$

$$z_{norm} = \begin{bmatrix} \frac{1}{\sigma_1} & 0 & 0 \\ 0 & \frac{1}{\sigma_2} & 0 \\ 0 & 0 & \frac{1}{\sigma_3} \end{bmatrix} \left(z - \begin{bmatrix} \mu_1 & \mu_1 & \mu_1 & \mu_1 \\ \mu_2 & \mu_2 & \mu_2 & \mu_2 \\ \mu_3 & \mu_3 & \mu_3 & \mu_3 \end{bmatrix} \right) = \begin{bmatrix} -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix}$$

۲.۵ ب

۱. شبکه‌ها سریع‌تر آموزش می‌بینند: علی رغم اینکه هر تکرار آموزش به دلیل محاسبات اضافی در جریان forward و پارامترهای اضافی برای آموزش در جریان backward بسیار کندتر می‌شود، اما همگرایی بسیار سریع‌تر رخ می‌دهد و به طور کلی فرآیند

آموزش سریع‌تر می‌شود.

۲. امکان استفاده از نرخ‌های یادگیری بالاتر: نزول گرادیان معمولاً نیازمند نرخ‌های یادگیری کوچکتر برای همگرایی شبکه است و هرچه شبکه‌ها عمیق‌تر می‌شوند، در جریان backward گرادیان‌های آنها کوچکتر می‌شوند. بنابراین نیاز به تکرارهای بیشتری دارند. استفاده از batch normalization امکان استفاده از نرخ‌های یادگیری بسیار بالاتر را فراهم می‌کند، که سرعت آموزش شبکه‌ها را بهبود می‌بخشد.

۳. سهولت در مقداردهی اولیه وزن‌ها: مقداردهی اولیه وزن‌ها ممکن است دشوار باشد و این مسئله در ساختارهای عمیق‌تر شدت می‌گیرد. batch normalization به ما امکان می‌دهد که نسبت به انتخاب وزن‌های اولیه کمتر حساسیت کمتری داشته باشیم.

۴. استفاده از توابع فعال‌سازی متنوع‌تر: برخی از توابع فعال‌سازی در برخی از مواقع به خوبی عمل نمی‌کنند. سیگمویدها به سرعت گرادیان خود را از دست می‌دهند. به این معنا که نمی‌توانند در شبکه‌های عمیق استفاده شوند. همچنین ReLUها اغلب در حین آموزش از بین می‌روند. به عبارتی کاملاً یادگیری متوقف شده است. بنابراین نیاز به توجه به بازه مقادیر وارد شده به آنها داریم. با تنظیم مقادیر وارد شده به هر تابع فعال‌سازی، batch normalization باعث می‌شود که توابع غیرخطی که به نظر می‌آید در شبکه‌های عمیق خوب کار نمی‌کنند، به صورت مجدد قابل استفاده شوند.

۵. سهولت در ساختاردهی شبکه‌های عمیق‌تر: به دلیل موارد گفته شده در بالا، ساختاردهی و آموزش شبکه‌های عصبی عمیق با استفاده از batch normalization ساده‌تر و سریع‌تر است و نشان داده شده است که شبکه‌های عمیق به طور کلی نتایج بهتری ارائه می‌دهند.

۶. regularization: batch normalization کمی نویز به شبکه اضافه می‌کند. در برخی موارد، مانند ماژول‌های Inception، batch normalization نیز همانند dropout عمل می‌کند. اما به طور کلی، batch normalization را به عنوان مقدار کمی regularization اضافی در نظر بگیرید که شاید امکان کاهش برخی از dropout را در یک شبکه فراهم کند.

۷. به طور کلی می‌تواند نتایج بهتری ارائه دهد: برخی آزمایش‌ها نشان می‌دهند که batch normalization به بهبود نتایج آموزش کمک می‌کند. با این حال این بهینه‌سازی به کمک آموزش سریع‌تر است، بنابراین نباید به آن به عنوان یک راه برای بهبود شبکه تلقی شود. اما از آنجا که اجازه می‌دهد تا شبکه‌ها را سریع‌تر آموزش بدهیم، این به این معناست که می‌توان به سرعت بیشتری روی طراحی‌های مختلف تست انجام بدهیم. همچنین این امکان را می‌دهد تا شبکه‌های عمیق‌تر بسازیم که به طور کلی بهتر هستند. پس احتمالاً نتایج بهتری خواهیم داشت.

۳.۵ ج

Covariate shift، وضعیتی است که در آن توزیع ویژگی‌های ورودی مدل در محیط تولید (تست) نسبت به آنچه مدل در طول آموزش و اعتبارسنجی دیده است تغییر می‌کند.

۴.۵ د

۶

۱.۶ الف

توابع فعال‌سازی واحد خطی تصحیح شده (ReLU) به چند دلیل به طور معمول در یادگیری عمیق بیشتر از توابع فعال‌سازی تانژانت هایپربولیک (tanh) مورد استفاده قرار می‌گیرند:

۱. مشکل محو شیب: توابع فعال‌سازی تانژانتی این خاصیت را دارند که مشتق آن‌ها در اطراف مقدار ۰ بیشترین مقدار را دارد. این به معنای آن است که در هنگام بازگشتی به عقب (backpropagation) ، زمانی که شیب محاسبه و به عقب از طریق شبکه منتقل می‌شود، شیب‌ها به اندازه‌ای کوچک می‌شوند که با افزایش مقدار مطلق ورودی، بسیار کوچک می‌شوند. این می‌تواند به مشکل محو شیب (vanishing gradient) منجر شود که در آن لایه‌های اولیه یک شبکه عمیق مقدار به‌روزرسانی کمی دریافت می‌کنند و به سختی می‌توانند به طور مؤثر یاد بگیرند. از طرف دیگر، تابع ReLU این مشکل محو شیب را ندارد زیرا مشتق آن برای ورودی‌های مثبت ۱ است.

۲. محاسبه ساده‌تر: تابع ReLU از نظر محاسباتی ارزان‌تر از تابع tanh است. محاسبه تابع tanh شامل توان‌گیری و تقسیم است، که عملیات محاسباتی مقایسه شده با عملیات آستانه‌گذاری ساده تابع ReLU بیشتر توان مصرف می‌کند. این باعث می‌شود که ReLU موثرتر و سریعتر برای آموزش باشد.

۳. sparsity و غیرخطیت: واحدهای ReLU می‌توانند sparsity در شبکه ایجاد کنند. زیرا برای ورودی‌های منفی خروجی ۰ می‌دهند. sparsity در برخی موارد مفید است. زیرا شبکه را تشویق می‌کند که بر روی زیرمجموعه‌ای از ویژگی‌های ورودی تمرکز کند. علاوه بر این، واحدهای ReLU غیرخطیت ارائه می‌دهند که برای مدل کردن توابع پیچیده توسط شبکه‌های عمیق مورد نیاز است.

۴. موفقیت تجربی: توابع ReLU در آموزش شبکه‌های عصبی عمیق موفقیت نشان داده‌اند. آن‌ها به طور گسترده در معماری‌های مختلف یادگیری عمیق مانند شبکه‌های عصبی کانوولوشنی (CNN) و شبکه‌های عصبی مکرر (RNN) به کار رفته‌اند و در مدل‌های برتر متعددی استفاده شده‌اند.

با این حال، مهم است به یاد داشت که تابع ReLU همراه با محدودیت‌های خود نیز هست. ممکن است مشکل "مرگ ReLU" رخ دهد که در آن واحدهای ReLU طی آموزش غیرفعال شوند (همیشه خروجی ۰ دهند) و هیچگاه به حالت عادی باز نگردند. این مشکل با استفاده از نسخه‌های تغییریافته از ReLU مانند Leaky ReLU یا Parametric ReLU قابل حل است که به واحدها امکان می‌دهد که برای ورودی‌های منفی شیبی کوچک داشته باشند و جلوی غیرفعال شدن کامل واحدها را بگیرند.

در عمل، انتخاب بین ReLU و tanh به مسئله خاص، معماری شبکه و سایر پارامترهای مدل بستگی دارد. در این مورد پاسخ یکتایی وجود ندارد و پژوهشگران اغلب با توابع فعال‌سازی مختلف آزمایش می‌کنند تا بیابند کدام تابع برای یک وظیفه خاص بهتر عمل می‌کند.

۲.۶ ب

یکی از مهم‌ترین موانع در آموزش شبکه‌های عصبی عمیق (DNN)، مشکل محو شیب است که در آن شیب‌ها (گرادیان‌ها) تابع هزینه نسبت به وزن‌های لایه‌های اولیه به‌طور قابل‌توجهی کوچک می‌شوند. به عبارت دیگر، لایه‌های اولیه کمتر یا اصلاً اطلاعات وزن به‌روزشده‌ای در هنگام بازگشت به عقب (backpropagation) دریافت می‌کنند که به همگرایی کند یا حتی رکود منتج می‌شود. مشکل محو شیب به‌طور اصلی به انتخاب توابع فعال‌ساز و روش‌های بهینه‌سازی در شبکه‌های عصبی عمیق مرتبط است.

روابط مشتق دو تابع به همراه نمودار مشتق آن‌ها رسم شده است.

6.2.1 ReLU

$ReLU(x) = x^+ = \max(0, x)$

$$ReLU'(x) = \begin{cases} 0, & \text{if } x < 0 \\ 1, & \text{if } x \geq 0 \end{cases}$$

6.2.2 Sigmoid

$\sigma(x) = \frac{1}{1+e^{-x}}$

$\sigma'(x) = \sigma(x)\,(1 - \sigma(x))$

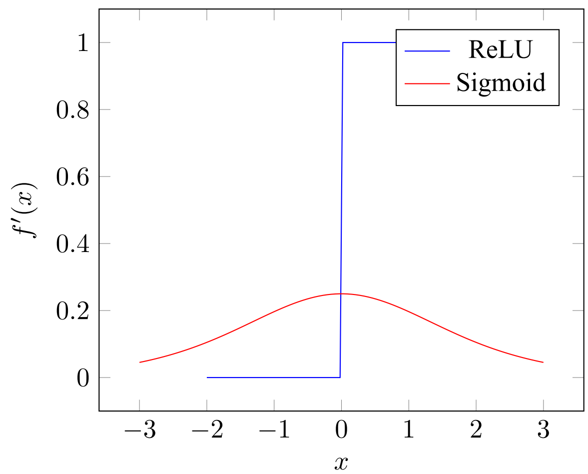


Figure 1: Derivatives of ReLU and Sigmoid functions.

شیب توابع فعال‌ساز سیگمویدی به طور معمول به سرعت محو می‌شوند. یک بازه نسبتاً کوچکی از ورودی‌ها وجود دارد که مشتق تابع سیگموید به میزان کافی غیرصفر است. به عبارت دیگر، هنگامی که سیگموید به یکی از دو قله چپ یا راست می‌رسد، تقریباً بی‌معنی است که یک مرور به عقب از آن انجام داده شود، زیرا مشتق به صفر میل می‌کند. از سوی دیگر، تابع ReLU تنها هنگامی اشباع می‌شود که ورودی کمتر از صفر باشد. حتی این اشباع می‌تواند با استفاده از واحدهای Leaky ReLU مهار شود. در شبکه‌های عمیق، اشباع یادگیری را مختل می‌کند، بنابراین ReLU می‌تواند راه‌حل مناسب‌تری نسبت به سیگموید باشد. البته باید به یاد داشت که با توجه به نوع داده و شرایط مسئله هر یک این توابع می‌تواند عملکرد بهتری نسبت به دیگری داشته باشد.

۳.۶ ج

وقتی مقدار اولیه وزن‌ها در یک شبکه عصبی با تابع فعال‌سازی Sigmoid بسیار بزرگ باشد، چند مشکل ممکن است پیش بیاید:

۱. اشباع شبکه: ورودی‌های بزرگ باعث می‌شود تابع فعال‌سازی Sigmoid به سرعت به ۱ (یا به سمت صفر) اشباع شود. این به این معناست که توابع فعال‌سازی Sigmoid به سرعت مقدارهای خروجی نزدیک به ۱ (یا ۰) تولید می‌کنند، و این می‌تواند به مشکل شبکه اصلی شما بیافزاید، زیرا گرادیان‌ها به سرعت به صفر نزدیک می‌شوند.

۲. مشکل محو شیب: همچنین، وقتی مقدار وزن‌ها بسیار بزرگ باشد، مشکل محو شیب (vanishing gradient) نیز ممکن است به وجود بیاید. زیرا مشتق تابع Sigmoid در نزدیکی نقطه میانی (۵.۰) بیشینه می‌شود و با افزایش فاصله از این نقطه در هر دو جهت، مشتق به سرعت به صفر نزدیک می‌شود. این موجب می‌شود که گرادیان‌ها بسیار کوچک شوند و به شبکه کمکی در آموزش نکنند.

۳. سختی آموزش: شبکه‌های عصبی با تابع فعال‌سازی Sigmoid و وزن‌های بزرگ ممکن است به سرعت به مسئله آموزش نشدنی تبدیل شوند. آموزش شبکه‌های ژرف با این ویژگی‌ها به طور کلی سخت‌تر است و نیازمند تنظیمات و حل مشکلات ویژه‌ای می‌باشد.

برای مقابله با این مشکلات، می‌توان از مقادیر اولیه وزن مناسب‌تری استفاده کرد، مانند مقادیر تصادفی کوچکتر یا از روش‌های مانند نرمالیزه کردن وزن‌ها بهره برد. همچنین، ممکن است از توابع فعال‌سازی دیگری که بهتر با مقادیر بزرگ کار می‌کنند، مانند ReLU یا Leaky ReLU، استفاده کرد. انتخاب تابع فعال‌سازی و مقادیر اولیه وزن‌ها باید به توجه به کاربرد و معماری خاص شبکه انجام شود.

۷

۱.۷ الف

در شبکه‌های عصبی، استفاده از توابع فعال‌ساز غیرخطی به دلایل متعددی ضروری است:

۱. قابلیت نمایش توابع پیچیده: توابع فعال‌ساز غیرخطی امکان نمایش توابع پیچیده‌تری را فراهم می‌کنند. اگر از توابع خطی استفاده شود (مثل تابع همانی)، شبکه توانایی نمایش توابع پیچیده و غیرخطی را نخواهد داشت. توابع غیرخطی می‌توانند الگوهای پیچیده‌تر و ساختارهای عمیق‌تر را مدل کنند.

۲. اهمیت انطباق به داده: توابع غیرخطی به شبکه‌های عصبی اجازه می‌دهند که بهتر به داده‌ها انطباق پیدا کنند. این به معنای این است که توابع فعال‌سازی غیرخطی می‌توانند بر اساس ویژگی‌های داده و نمونه‌ها، تغییر کنند و توانایی شبکه را در تطبیق بهتر با تغییرات در داده ارتقا می‌دهند.

۳. مدل‌سازی تعاملات غیرخطی: در بسیاری از کاربردها، ارتباطات و تعاملات بین متغیرها و ویژگی‌های ورودی غیرخطی هستند. توابع فعال‌سازی غیرخطی به شبکه‌ها امکان می‌دهند تا تعاملات غیرخطی را به خوبی مدل کنند و از توانایی شبکه در پیش‌بینی تعاملات پیچیده بهره‌برند.

۴. مشکل محو شیب را کاهش می‌دهند: توابع غیرخطی معمولاً مشکل محو شیب را کاهش می‌دهند. توابع خطی به سرعت به صفر همگرا می‌شوند و مشتق‌های کوچکی دارند، اما توابع فعال‌سازی غیرخطی (مانند ReLU) مشتق‌های بزرگ‌تری دارند و این به شبکه‌ها امکان می‌دهد که در دوره‌های آموزشی عمیق‌تر پایدارتر باشند.

با توجه به این دلایل، توابع فعال‌سازی غیرخطی مهمی در موفقیت شبکه‌های عصبی در مدل‌سازی و پیش‌بینی وظایف پیچیده و غیرخطی ایفا می‌کنند.

۲.۷ ب

$$g\left(x\right)=-\min\left(5,x\right)$$

$$h\left(x\right)=\begin{cases}\max\left(x,0.3x\right), & x\geq 0\\ \min\left(x,0.3x\right), & x<0\end{cases}$$

$$\xRightarrow[\text{if } x<0:\ x<0.3x]{\text{if } x\geq 0:\ x\geq 0.3x} h\left(x\right)=x$$

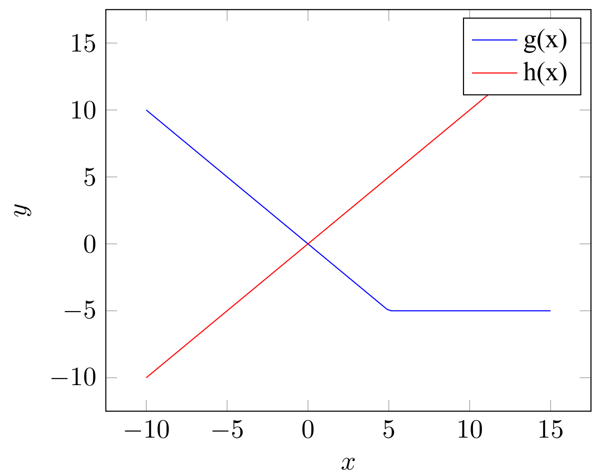


Figure 2: Plot of $g(x)=-\min(5,x)$ and $h(x)=x$

توابع فعال‌ساز نقش حیاتی در شبکه‌های عصبی عمیق دارند و تأثیر زیادی بر توانایی شبکه در یادگیری و حل مسائل مختلف دارند. انتخاب توابع فعال‌ساز می‌تواند به طور قابل توجهی بر آموزش و عملکرد شبکه تأثیر بگذارد.

۱.۲.۷ $g(x)$

این تابع حداقل مقدار بین x و ۵ را می‌گیرد، آن را منفی می‌کند و نتیجه را به عنوان خروجی استفاده می‌کند. این تابع می‌تواند به عنوان تابع فعال‌سازی استفاده شود، اما مشکلاتی دارد:

۱. اشباع: این تابع برای مقادیر $x>5$ صاف می‌شود و گرادیان آن در این ناحیه صفر است. این می‌تواند منجر به محو شدن گرادیان‌ها در هنگام backpropagation شود، که باعث دشوار شدن آموزش شبکه می‌شود.

۲. عدم غیرخطیت: توابع فعال‌سازی باید غیرخطیت را به وارد شبکه کنند. $g(x)$ یک تابع قطعه قطعه خطی است که تنها یک نقطه انحراف در $x=5$ دارد. این کمبود غیرخطیت می‌تواند توانایی شبکه در مدل کردن روابط پیچیده در داده‌ها را محدود کند.

۳. عدم محبوبیت: این تابع در عمل برای شبکه‌های عصبی عمیق به صورت معمول مورد استفاده قرار نمی‌گیرد، و توابع فعال‌سازی معتبرتری وجود دارند که به خوبی کار می‌کنند، مانند ReLU و نسخه‌های مشتق شده از آن.

۲.۲.۷ $h(x)$

این تابع یک تابع فعال‌سازی خطی است، یعنی غیرخطیتی را به شبکه معرفی نمی‌کند. اگرچه ممکن است در برخی معماری‌های شبکه‌های عصبی (مانند رگرسیون خطی) مورد استفاده قرار گیرد، اما به طور کلی در شبکه‌های عصبی عمیق به عنوان تابع فعال‌سازی

استفاده نمی‌شود، به علت دلایل متعددی از جمله:

۱. ظرفیت مدل‌سازی محدود: شبکه‌های عصبی عمیق از توابع فعال‌سازی غیرخطی برای مدل‌سازی روابط غیرخطی و پیچیده در داده‌ها استفاده می‌کنند. استفاده از $h(x) = x$ در تمام شبکه تقریباً شبیه به یک مدل خطی تک لایه می‌شود و توانایی نمایش الگوهای پیچیده را محدود می‌کند.

۸

۱.۸ الف

تابع تانژانت هایپربولیک ($tanh$) اغلب به عنوان نسخه مقیاس‌شده‌ای از تابع سیگموید توصیف می‌شود، به خصوص تابع سیگموید لجستیک. این رابطه به دلیل شباهت‌های تابع تانژانت و تابع سیگموید وجود دارد، اما در بازه و مقیاس‌شان تفاوت دارند.
تابع سیگموید که اغلب با نماد $\sigma(x)$ نشان داده می‌شود، به شرح زیر تعریف می‌شود:

$\sigma(x) = \frac{1}{1+e^{-x}}$

این تابع هر عدد حقیقی را به یک مقدار بین ۰ و ۱ نگاشت می‌کند. وقتی x یک عدد مثبت بزرگ است، $\sigma(x)$ به ۱ نزدیک می‌شود و وقتی x یک عدد منفی بزرگ است، $\sigma(x)$ به ۰ نزدیک می‌شود. این به این معناست که تابع سیگموید ورودی خود را در بازه (۰، ۱) فشرده می‌کند که برای مسائل دسته‌بندی دودویی مفید است، چون می‌توان از آن تعبیر احتمالاتی کرد.
تابع تانژانت هایپربولیک، به شکل زیر تعریف می‌شود:

$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} = \frac{e^{2x}-1}{e^{2x}+1}$

تابع تانژانت هایپربولیک هر عدد حقیقی را به یک مقدار بین ۱- و ۱ نگاشت می‌دهد. وقتی x یک عدد مثبت بزرگ است، $tanh(x)$ به ۱ نزدیک می‌شود و وقتی x یک عدد منفی بزرگ است، $tanh(x)$ به ۱- نزدیک می‌شود.

رابطه بین توابع تانژانت هایپربولیک و سیگموید به شرح زیر است:

۱. مقیاس‌دهی: تابع تانژانت هایپربولیک، انتقال داده شده و مقیاس شده‌ی تابع سیگموید به منظور داشتن رنج $(-1, 1)$ به جای $(0, 1)$ است. این مقیاس‌دهی با کم کردن 0.5 از تابع سیگموید و سپس ضرب آن در ۲ انجام می‌شود:

$2\sigma(2x) - 1 = 2 \times \frac{1}{1+e^{-2x}} - 1 = \frac{1-e^{-2x}}{1+e^{-2x}} = \frac{1-e^{-2x}}{1+e^{-2x}} \times \frac{e^{2x}}{e^{2x}} = \frac{e^{2x}-1}{e^{2x}+1} = tanh(x)$
$\Rightarrow tanh(x) = 2\sigma(2x) - 1$

۲. تقارن: یکی از تفاوت‌های کلیدی این است که تابع تانژانت هایپربولیک در اطراف مبدأ متقارن است (در واقع $tanh(-x) = -tanh(x)$)، درحالی که تابع سیگموید این تقارن را ندارد.

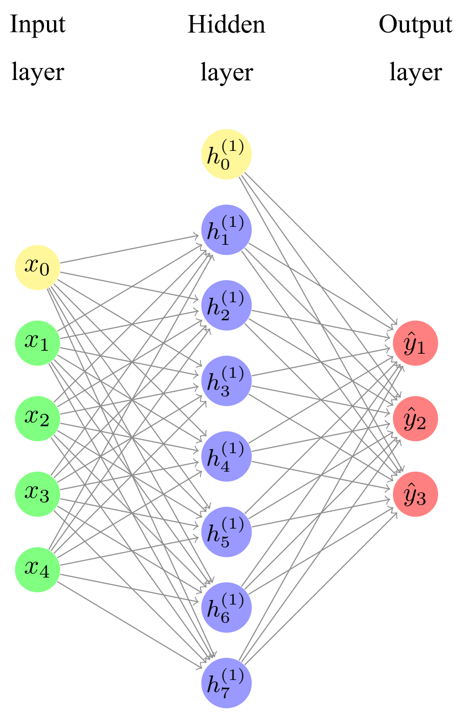
تابع تانژانت هایپربولیک اغلب به عنوان یک جایگزین برای تابع سیگموید در شبکه‌های عصبی استفاده می‌شود. زیرا به دلیل ماهیت مرکز شده حول صفری که دارد، یادگیری در شبکه‌های عمیق را سریع‌تر می‌کند. اینکه تابع تانژانت هایپربولیک مقادیر منفی را نیز خروجی دهد به معنای این است که می‌تواند تغییرات مثبت و منفی را در واحدهای مخفی شبکه در طول آموزش ایجاد کند که می‌تواند به همگرایی کمک کند. با این حال، هر دو تابع هنوز در متنوعی از زمینه‌ها استفاده می‌شوند و انتخاب بین آن‌ها بستگی به مسئله خاص و معماری شبکه دارد.

۲.۸ ب

$p\left(x\right) = x\log\left(1 + tanh\left(e^{x}\right)\right)$

$$\frac{d}{dx}p\left(x\right) = \left(\frac{d}{dx}x\right) \times \log\left(1 + tanh\left(e^{x}\right)\right) + \left(\frac{d}{dx}\left(\log\left(1 + tanh\left(e^{x}\right)\right)\right)\right) \times x$$
$$= \log\left(1 + tanh\left(e^{x}\right)\right) + x\left(\frac{d}{dx}\left(1 + tanh\left(e^{x}\right)\right)\right) \times \frac{1}{1+tanh(e^{x})}$$
$$= \log\left(1 + tanh\left(e^{x}\right)\right) + \frac{x}{1+tanh(e^{x})} \times \left(\frac{d}{dx}\left(tanh\left(e^{x}\right)\right)\right)$$
$$= \log\left(1 + tanh\left(e^{x}\right)\right) + \frac{x}{1+tanh(e^{x})} \times \left(\frac{d}{dx}\left(e^{x}\right)\right) \times \left(1 - tanh^{2}\left(e^{x}\right)\right)$$
$$= \log\left(1 + tanh\left(e^{x}\right)\right) + xe^{x}\left(1 - tanh\left(e^{x}\right)\right)$$

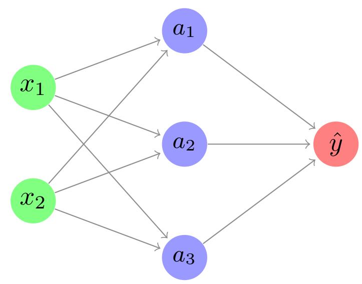
۹



با توجه به شبکه‌ی بالا، نورون‌های سبز، بنفش، و قرمز به ترتیب لایه‌ی ورودی، لایه‌ی مخفی، و لایه‌ی خروجی را تشکیل می‌دهند و همچنین نورون‌های زرد biasها هستند که همگی مقدار ۱ دارند. پارامترهای قابل یادگیری شبکه وزن‌های موجود بین نورون‌هاست که تعدادشان برابر است با: $4 \times 7 + 7 + 7 \times 3 + 3 = 59$

۱۰

Input Layer | ReLU Layer | Output Layer



$$J(W) = \frac{1}{n}\sum_{i=1}^{n} L\left(\hat{y}^{(i)}, y^{(i)}\right) = \left(\hat{y}^{(1)} - y^{(1)}\right)^2 = (\hat{y} - 3)^2$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \\ w_5 & w_6 \end{bmatrix} x$$

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = ReLU(z)$$

$$z_4 = \begin{bmatrix} w_7 & w_8 & w_9 \end{bmatrix} a$$

$$\hat{y} = ReLU(z_4)$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$y = 3$$

$$\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & -2 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 4 \\ -3 \\ 5 \end{bmatrix}$$

$$a = ReLU\left(\begin{bmatrix} 4 \\ -3 \\ 5 \end{bmatrix}\right) = \begin{bmatrix} 4 \\ 0 \\ 5 \end{bmatrix}$$

$$z_4 = \begin{bmatrix} -1 & 3 & 2 \end{bmatrix} \begin{bmatrix} 4 \\ 0 \\ 5 \end{bmatrix} = 6$$

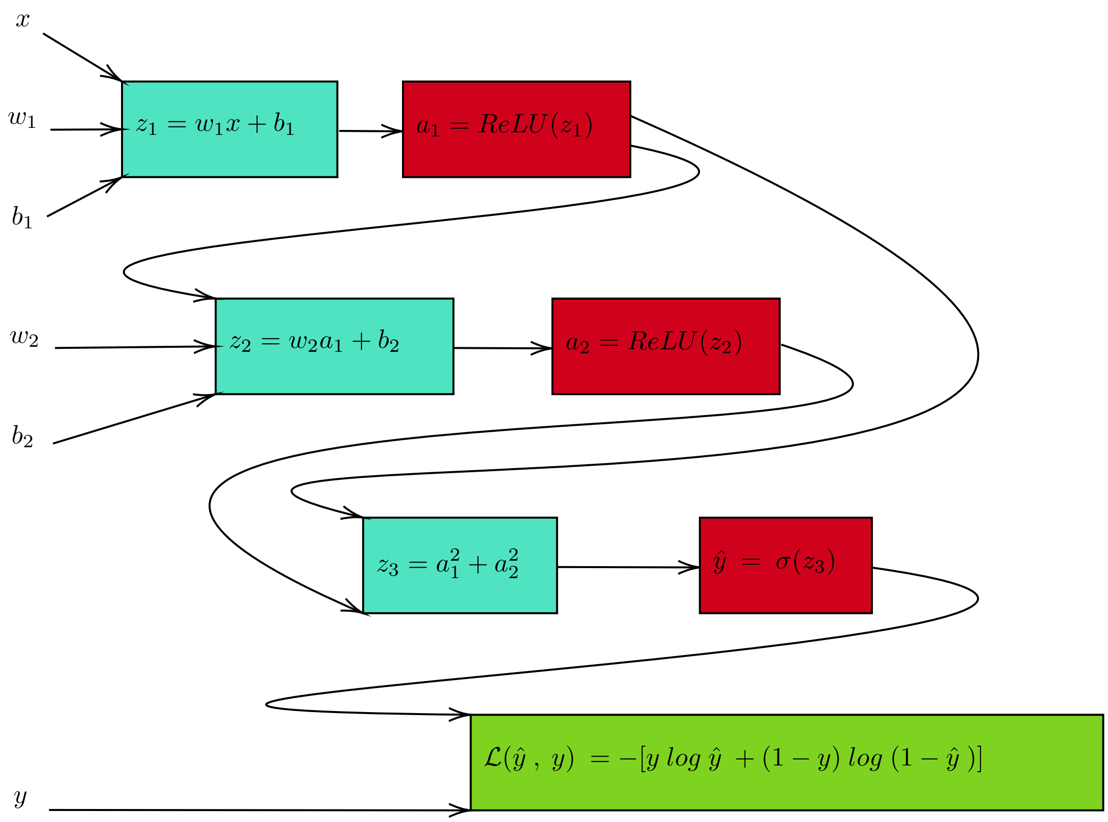
$$\hat{y} = ReLU(6) = 6$$

$$
\begin{aligned}
&\begin{bmatrix} w_7 & w_8 & w_9 \end{bmatrix} \leftarrow \begin{bmatrix} w_7 & w_8 & w_9 \end{bmatrix} - \alpha \begin{bmatrix} \frac{\partial J(W)}{\partial w_7} & \frac{\partial J(W)}{\partial w_8} & \frac{\partial J(W)}{\partial w_9} \end{bmatrix} \\
&= \begin{bmatrix} -1 & 3 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_7} & \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_8} & \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_9} \end{bmatrix} \\
&= \begin{bmatrix} -1 & 3 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} 2\left(\hat{y} - y\right) a_1 & 2\left(\hat{y} - y\right) a_2 & 2\left(\hat{y} - y\right) a_3 \end{bmatrix} \\
&= \begin{bmatrix} -1 & 3 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} 24 & 0 & 30 \end{bmatrix} = \begin{bmatrix} -3.4 & 3 & -1 \end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
&\begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \\ w_5 & w_6 \end{bmatrix} \leftarrow \begin{bmatrix} w_1 & w_2 \\ w_3 & w_4 \\ w_5 & w_6 \end{bmatrix} - \alpha \begin{bmatrix} \frac{\partial J(W)}{\partial w_1} & \frac{\partial J(W)}{\partial w_2} \\ \frac{\partial J(W)}{\partial w_3} & \frac{\partial J(W)}{\partial w_4} \\ \frac{\partial J(W)}{\partial w_5} & \frac{\partial J(W)}{\partial w_6} \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 \\ 1 & -2 \\ 1 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1} & \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_2} \\ \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_3} & \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_4} \\ \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \times \frac{\partial z_3}{\partial w_5} & \frac{\partial J(W)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial a_3} \times \frac{\partial a_3}{\partial z_3} \times \frac{\partial z_3}{\partial w_6} \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 \\ 1 & -2 \\ 1 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} 2\left(\hat{y} - y\right) \times w_7 \times 1 \times x_1 & 2\left(\hat{y} - y\right) \times w_7 \times 1 \times x_2 \\ 2\left(\hat{y} - y\right) \times w_8 \times 0 \times x_1 & 2\left(\hat{y} - y\right) \times w_8 \times 0 \times x_2 \\ 2\left(\hat{y} - y\right) \times w_9 \times 1 \times x_1 & 2\left(\hat{y} - y\right) \times w_9 \times 1 \times x_2 \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 \\ 1 & -2 \\ 1 & 2 \end{bmatrix} - 0.1 \begin{bmatrix} -6 & -12 \\ 0 & 0 \\ 12 & 36 \end{bmatrix} = \begin{bmatrix} 2.6 & 2.2 \\ 1 & -2 \\ -0.2 & -1.6 \end{bmatrix}
\end{aligned}
$$

۱۱

۱.۱۱ الف



۲.۱۱ ب

$$\frac{\partial L}{\partial a_1} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_3} \times \frac{\partial z_3}{\partial a_1} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) \times \hat{y}\,(1-\hat{y}) \times 2a_1 = (\hat{y}-y)\,2a_1$$

$$\frac{\partial L}{\partial a_2} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_3} \times \frac{\partial z_3}{\partial a_2} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) \times \hat{y}\,(1-\hat{y}) \times 2a_2 = (\hat{y}-y)\,2a_2$$

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_3} \times \left(\frac{\partial z_3}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1} + \frac{\partial z_3}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}\right)$$
$$= \frac{\partial L}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1} + \frac{\partial L}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial a_1} \times \frac{\partial a_1}{\partial z_1} \times \frac{\partial z_1}{\partial w_1}$$
$$= (\hat{y}-y)\,2a_1 \times a_1\,(1-a_1) \times x + (\hat{y}-y)\,2a_2 \times a_2\,(1-a_2) \times w_2 \times a_1\,(1-a_1) \times x$$
$$= 2\,(\hat{y}-y)\,a_1\,(1-a_1)\,x\left[a_1 + a_2^2\,(1-a_2)\,w_2\right]$$

$$\frac{\partial L}{\partial w_2} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_3} \times \frac{\partial z_3}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_2}$$
$$= \frac{\partial L}{\partial a_2} \times \frac{\partial a_2}{\partial z_2} \times \frac{\partial z_2}{\partial w_2} = (\hat{y}-y)\,2a_2 \times a_2\,(1-a_2) \times a_1$$

منابع

[1] https://jaykmody.com/blog/stable-softmax/

[2] https://towardsdatascience.com/batch-normalization-8a2e585775c9