پاسخ نامه تمرین تئوری سوم، شبکه های عصبی

بخش اول – مباحث تئوری و مسائل تشریحی

۱- گفته می‌شود که ایده شبکه های منصوعی عصبی از نحوه عملکرد سلول های عصبی مغز الهام گرفته شده است. اما نورون های مدل سازی شده با نورون های مغز تفاوت های اساسی دارند.

- چه چیزی نورون‌های بیولوژیکی مغز را متفاوت می‌کند؟ مهم ترین تفاوت هارا نام ببرید.

مهم‌ترین تفاوت ها شامل:

تعداد نورون ها، در مغز بسیار بیشتر از شبکه های عصبی مصنوعی است همچنین اتصالات در بین نورون های مغز خیلی بیشتر و پیچیده‌تر از شبکه های عصبی معمولی است. (هرچند در مواردی می‌توان شبکه های بازگشتی و پیچیده طراحی کرد اما این سطح از پیچیدگی معمولا مدل مناسبی را نتیجه نمی‌دهد) نورون های بیولوژیکی برای انتقال داده بین یکدیگر علاوه بر سیگنال های الکتریکی، از سیگنال های شیمیایی هم استفاده می کنند. سرعت تبادل اطلاعات در مغز پایین تر است اما مقاومت آنها نسبت به سیگنال های نادقیق بیشتر است. یادگیری در مغز به نحو متفاوتی انجام می‌شود و باعث می‌شود که لزوما به داده های زیادی برای آموزش نیاز نباشد. از لحاظ زیست شناختی نورون های مغز تا حدودی توانایی ترمیم خود را دارند، همچنین اگر بخشی از نورون های مغز از کار بیوفتد کل عملکرد مغز دچار فروریزش نمی‌شود که نشان دهنده وجود نوعی استقلال در بخش های مختلف آن است.

- در دهه های اخیر، با پیشرفت توان پردازشی کامپیوترها، ساخت و آموزش شبکه های عصبی مصنوعی در ابعاد هرچه بزرگتر میسر شده است. (برای مثال مدل GPT-3 دارای ۱۷۵ میلیارد پارامتر آموزش داده شده است) با کمی تحقیق این عدد را با تعداد سیناپس های مغز انسان مقایسه کنید. به نظر شما رسیدن به تعداد نورون های یک مغز واقعی ، عملکرد در سطح آن را هم تضمین می‌کند؟

مطالعات از وجود ۱۰۰ میلیارد نورون و حدود ۱۵هزار اتصال به ازای هر نورون (= تعداد وزن هایی از مرتبه ۱۰۰ الی ۱۰۰۰ ترلیون در شبکه های عصبی مصنوعی) در مغز گزارش می‌دهند. که با رقم فعلی شبکه های عصبی مصنوعی قابل مقایسه نیست. با این وجود افزایش تعداد نورون ها **الزاما** به معنی رسیده به همان سطح هوشمندی و کارکرد هم نیست چرا که برای مثال تعداد نورون های مغز یک قورباغه ۱۶ میلیون است و تعداد اتصالات آن از مرتبه چند میلیارد خواهد بود (هم مرتبه *GPT-3*) با این حال نمی‌توان ادعا کرد که هوشمندی این مدل از همه لحاظ در حد هوشمندی مغز قورباغه است.

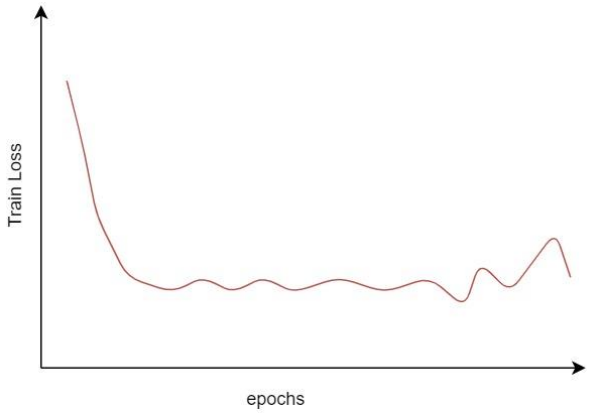
- در بین انواع روش های نوین یادگیری ماشین، ***یادگیری تقویتی*** شباهت زیادی با نحوه تعامل موجودات زنده با محیط را دارد. در خصوص نحوه کار این روش توضیح دهید.

این روش بر پایه یک نظام پاداش و خطا عمل می‌کند. یک عامل هوشمند با مجموعه اعمالی که مجاز است انجام دهد در یک محیط قرارداده می‌شود و عامل به ازای هر تعامل با محیط ، پاداش و مجازاتی دریافت می‌کند. تلاش عامل بیشینه کردن پاداش و کمینه کردن مجازات ها است و لذا با تعریف صحیح این توابع، عامل می‌تواند رفتارها و سلسله اعمالی که بیشترین سود را حاصل کند را خود به خود و بدون آموزش بیرونی یا دادن داده برچسب خورده یاد بگیرد.

- الگوریتم backpropagation عامل اصلی *یادگیری* در شبکه های عصبی محسوب می‌شود. اما شواهدی دال بر وجود چنین الگوریتمی در مغز وجود ندارد. به نظر شما دلیل آن چیست؟

دلیل آن این است که وجود این الگوریتم در مغز توجیهی ندارد. در واقع سرعت انتقال سیگنال ها در مغز این اجازه را نمی‌دهند که یادگیری ازین طریق رخ دهد. همچنین همه نورون های مغز دقیقا یکسان نیستند (برخلاف وزن ها که اعدادی اعشاری و عموما *float64* هستند) و نمی‌توان با یک روش همه آنهارا پیمایش کرد. هرچند دانش ما در خصوص نحوه یادگیری مغز هنوز کامل نیست، اما شواهد نشان می‌دهند که یادگیری در مغز با نوعی *ایجاد و تقویت مسیر های عصبی* همراه است. هر زمان که یک مسیر عصبی روشن شود (در اثر مشاهده، انجام عملی، دریافت احساسی و...) یادگیری صورت می‌پذیرد و هرچه این مسیر های عصبی بیشتر روشن شوند، یادگیری هم بهتر و ماندگار تر است.

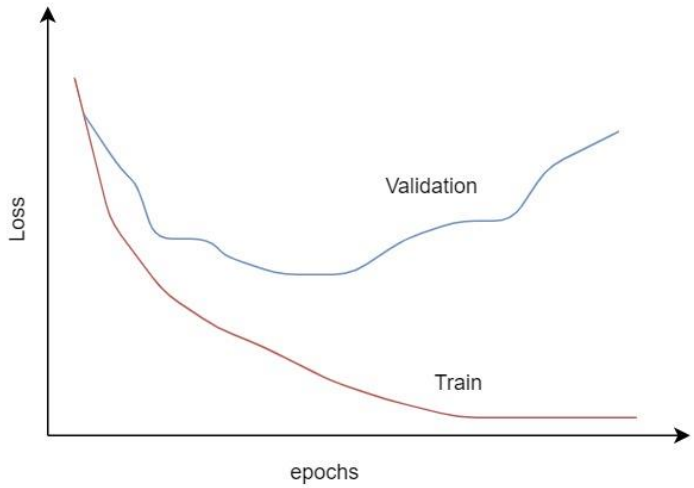
۲- علی می‌خواهد با داده های مربوط به ساعات عبور و مرور افراد در آزمایشگاه، یک مدل شبکه عصبی پرسپترون ۲ لایه ای را آموزش دهد تا بتواند در هر ساعت تعداد افراد حاضر در آزمایشگاه را پیش‌بینی کند. او پس از آموزش روی داده های $train$ نمودار خطا را بعد از هر $epoch$ رسم کرد و شکل زیر حاصل شد:



- چه مشکلی در هایپرپارامتر های مدل وجود دارد؟ لااقل یک پیشنهاد برای رفع مشکل بدهید.

اتفاقی که رخ داده آن است که پارامتر های مدل همگرا نمی‌شوند. (حول نقطه بهینه نوسان می کنند) اولین دلیلی که به ذهن می‌رسد بیش از اندازه بزرگ بودن نرخ یادگیری است. بهتر است نرخ یادگیری را کوچکتر کنیم و دوباره امتحان کنیم.

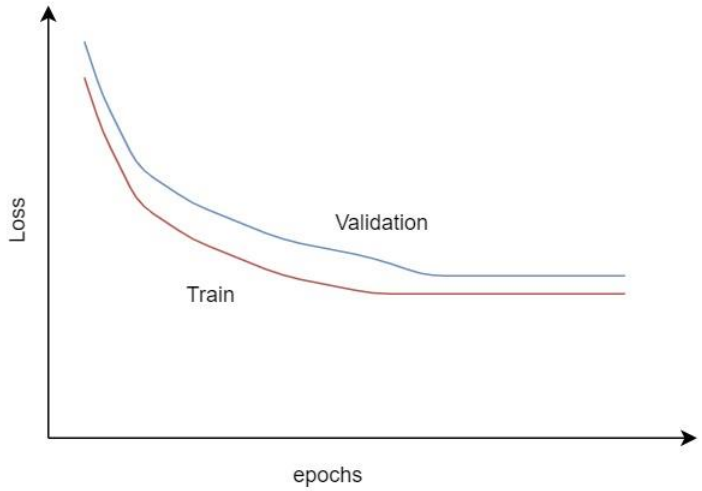
علی پس از تلاش زیاد توانست نمودار خطا روی داده های آموزش را نزدیک به صفر برساند. اما خوشحالی اش با رسم نمودار خطای داده های $validation$ فروکش کرد:



- مشکل مدل علی چه چیزی است؟ برای رفع این مشکل چه پیشنهادی دارید ؟ (لااقل ۳ مورد)

مشکلی که رخ داده بیش‌برازش (*overfitting*) نام دارد، یعنی مدل روی داده آموزشی خیلی خوب عمل می‌کند اما بر روی داده تست خوب عمل نمی‌کند. کار هایی که می‌تواند بیش‌برازش را کاهش دهد: ۱- کاهش پیچیدگی مدل (کاهش تعداد نورون ها و لایه ها) ۲- افزایش داده آموزشی ۳- حذف تعدادی از شاخصه (*feature*) ها ۴- استفاده از *regularization* و...

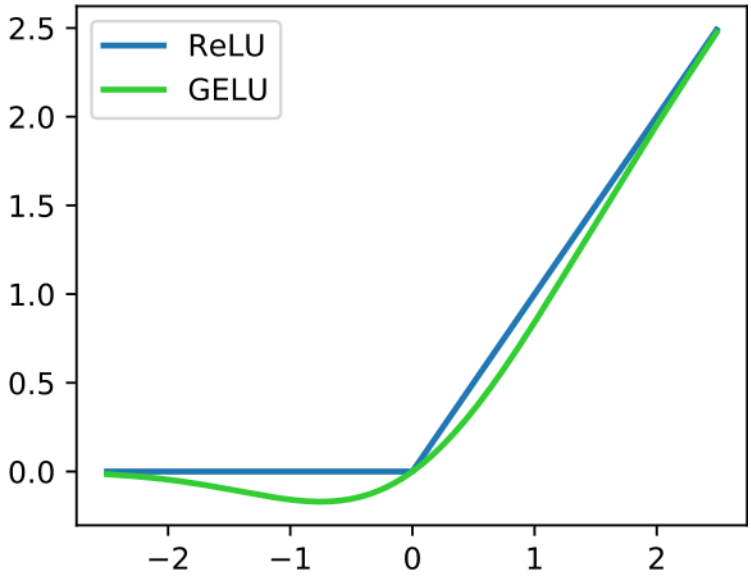
علی پس از مشورت با استاد درس هوش محاسباتی خود، تصمیم گرفت تغییراتی در مدل خود انجام دهد. به طور دقیق تر او **تعداد نورون ها** و **لایه** های مدل را تغییر داد. نمودار زیر خطای *Train* و *validation* را بعد از تغییرات علی را نشان می دهد.



- به نظر شما علی چه تغییری در مدل اعمال کرده بود؟ دلیل خود را بیان کنید.

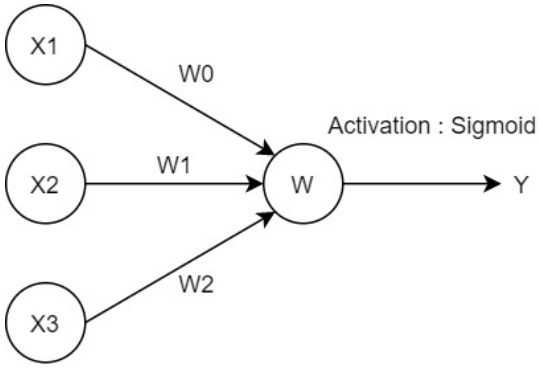
اتفاقی که افتاده آن است که خطای آموزش و ولیدیشن هردو کاهش پیدا کرده اند (که نشانه خوبی است) اما از حدی پایین تر نیامده اند. این مشکل در واقع *underfitting* نام دارد. یعنی مدل برای داده های آموزش بیش از اندازه ساده است و نمی‌تواند همه ویژگی های داده هارا لحاظ کند و در نتیجه در پیشبینی برچسب ها دقت کافی ندارد. لذا احتمالا علی لایه ها و نورون هارا خیلی کاهش داده است.

۳- تا چند سال پیش تابع *sigmoid* معمول ترین گزینه برای اعمال رفتار غیرخطی در خروجی نورون ها بود، اما در سال های اخیر و با عمیق تر شدن مدل ها، جایش را به توابعی مثل *relu* و *gelu* داده است. با رسم شکل هر کدام ازین توابع، دلیل این جایگزینی را توضیح دهید.



دلیل اصلی روی آوردن به این توابع این است که با عمیق شدن شبکه ها، مجبور هستیم برای محاسبه گرادیان ها، مشتقات توابع *activation* زیادی را پشت سر هم ردیف کنیم (طبق قاعده زنجیره ای) و به دلیل آنکه تابع *sigmoid* همه خروجی هارا بین ۰ تا ۱ می‌برد، عدد های گرادیان ها بعد از محاسبه تعدادی لایه به صفر می‌رسند و لذا خطای محاسبه شده تا وقتی به لایه های اول برسد عملا چیزی از آن نمانده و آموزش در لایه های ابتدایی رخ نمی‌دهد (یا خیلی دیر رخ می‌دهد). این مشکل که به اسم **گرادیان‌های محو شونده** (***vanishing gradient***) معروف است در صورت استفاده از توابع بالا کمرنگ‌تر می شود چرا که خروجی مشتق توابع *activation* متناسب با میزان خطا خواهد بود و خصوصا در اوایل الگوریتم که خطا بالا است، اثر خودش را خیلی سریع تر به لایه های اول مدل می‌رساند.

۴- می‌خواهیم یک شبکه پرسپترون تک لایه با یک نورون را آموزش دهیم.



اگر تابع هزینه ما

باشد $cost(x) = (sigmoid(Wx + b) - y_t)^2$

(W بردار وزن‌ها ، b عدد بایاس و y_t برچسب داده ورودی است)

موارد زیر را به دست آورید

- ابتدا مشتق تابع ***cost*** نسبت به ***W0*** را به دست آورید. (از قاعده زنجیره ای کمک بگیرید)

$$\frac{dcost}{dW_0} = 2X_1 sigmoid'(WX+b)(sigmoid(WX+b)-y_t)$$

$$= 2X_1 sigmoid(WX+b)(1-sigmoid(WX+b))(sigmoid(WX+b)-y_t)$$

- فرض کنید مقدار اولیه وزن ها به صورت ***w0 =0.5, w1 =0.2, w2 =0.3, b =2*** باشد. درین صورت مقدار خروجی مدل (***feed forward***) به ازای ورودی $\{۰،۱،۰\}$ و برچسب : ۲، و خطای به دست آمده را حساب کنید.

$$WX = [0.5 \quad 0.2 \quad 0.3]\begin{bmatrix}0\\1\\0\end{bmatrix} = 0.2, \quad WX+b = 2.2$$

$$Y = sigmoid(WX+b) = sigmoid(2.2) = \mathbf{0.9}$$

$$cost = (0.9-2)^2 = 1.21$$

- با استفاده از مشتقات به دست آمده در قسمت اول و داده های قسمت دوم، یک بار وزن ***W0*** را به‌روزرسانی کنید.

(طبق روش ***gradient decent*** و نرخ یادگیری = ۰،۱)

$$W_{0\,new} = W_{0\,old} - \alpha\frac{dcost}{dW_0} = 0.5 - 2X_1 S(WX+b)\big(1-S(WX+b)\big)(S(WX+b)-y_t)$$

$$0.5-(0.1)\,(2)\,(0)\,(0.9)(1-0.9)(0.9-2) = 0.5$$

موفق باشید

تیم تدریسیاری