تکلیف دوم

مجتبی ملائی
۴۰۱۳۱۳۸۳

۱

۱. خیر لزوما نمی‌تواند مشکل بیش برازش را از بین ببرد. زیرا داده‌های جدید باید اطلاعات جدیدی را به شبکه اضافه کنند. اگر این داده ها متفاوت نباشند و شباهت زیادی با داده های قبلی داشته باشند باعث می‌شود که اطلاعات جدیدی به شبکه وارد نشود، پس نمی‌تواند مشکل اورفیت را حل کند.

۲. درست است. وقتی مدل روی دیتای train بسیار خوب و دقیق عمل می‌کند اما روی داده های تست دقت کمی دارد یعنی مدل روی داده های train اورفیت شده است و روی داده های واقعی و خارج از train خوب کار نمی‌کند.

۳. درست است. زیرا در $\mathrm{L1}$ عبارت λ را به dW اضافه می‌کنیم که مشتق $\lambda \sum_{i=0}^{n} |W_i|$ است. درمقابل $\mathrm{L2}$ عبارت λW_i را به dW اضافه می‌کند که مشتق $\frac{\lambda}{2} \sum_{i=0}^{n} W_i^2$ است. در $\mathrm{L2}$ مشتق وابسته به W می‌شود که در واقع با کوچک شدن W این عبارت نیز تا حد زیادی کوچک می‌شود و هرچقدر این مقدار کوچک تر شود، dW نیز کمتر کوچک می‌شود و باعث می‌شود سرعت کاهش W کمتر شود و مقدار آن به صفر نرسد. درحالی که در $\mathrm{L1}$ همواره به یه مقدار ثباتی کم می‌شود تا به صفر برسد.

۴. خیر inverted dropout نیز همانند dropout فقط در مرحله train انجام می‌شود تا جلوی overfit شدن را بگیرد. اما تفاوت آن این است که در train یک scaling به اندازه $\frac{1}{p}$ که در آن p احتمال نگه‌داشتن نورون است، انجام می‌شود. اما در مرحله استنتاج باید dropout خاموش شود. چون اگر استفاده کنیم مدل هربار نتایج مختلفی خواهد داد بنابراین استفاده از آن در استنتاج منطقی نخواهد بود.

۵. خیر از early stopping برای جلوگیری از over-fitting استفاده می‌شود نه برای بهینه سازی hyper-parameter ها.

۶. درست است. چون وقتی اندازه mini-batch برابر با یک باشد، فقط یک داده را هر بار می‌بیند و وزن ها را بعد از دیدن هر داده آپدیت می‌کند که این همان گرادیان نزولی تصادفی است.

۷. نادرست است. نرمال سازی باعث تسریع و هموار شدن(smooth) راه آموزش میشود و سرعت آن را افزایش می‌دهد. همچنین هدف نرمال سازی فرار از بهینه محلی نیست اما شاید به صورت غیرمستقیم در آن تاثیر گذار باشد.

۸. خیر همیشه نمی‌تواند کافی باشد. زیرا ممکن است توازن داده ها رعایت نشود. به عنوان مثال اگر بیشتر داده ها از کلاس A هستند و مقدار کمی از B باشند، اگر به صورت تصادفی انتخاب شوند ممکن است یکی از مجموعه ها هیچ داده ای از B نداشته باشد. همچنین در برخی داده ها ترتیب زمانی مهم است و نمی‌توان به صورت تصادفی انتخاب کرد.

۹. خیر به وزن های گذشته به صورت غیر خطی (نمایی) مقدار می‌دهد. هرچقدر وزن قدیمی تر باشد مقدار آن به صورت نمایی کاهش می‌یابد.

۲

۱. تابع فعال‌ساز tanh نسبت به sigmoid در لایه‌های میانی ترجیح داده می‌شود زیرا خروجی آن در بازه $[-1, 1]$ و دارای میانگین صفر است (zero-centered). این خاصیت موجب می‌شود که از تجمع گرادیان در یک سمت (مثلاً مثبت در sigmoid) جلوگیری شود. در نتیجه، tanh باعث بهبود انتشار گرادیان، همگرایی سریع‌تر شبکه (به دلیل مشتق بیشتر) و کاهش احتمال بروز مشکل vanishing gradient نسبت به sigmoid می‌شود.

۲. زیرا در این زمان به دلیل غیر فعال شدن تصادفی برخی نورون ها تابع loss ناپایدار و نویز دار است و مقدار آن قابل تحلیل دقیق نیست.

۳. در این روش مقدار های قبلی وزن ها نیز در نظر گرفته می‌شود بنابراین به‌حای اینکه فقط داده پر نویز لحظه ای در نظر گرفته شود، روند کلی داده ها در نظر گرفته خواهد شد. در این حالت نویز ها تا حد خوبی حذف می‌شوند بنابراین باعث می‌شود تا الگوی صاف تری دیده شود.

۴. شبکه‌های عصبی عمیق نیاز به مقداردهی اولیه دارند تا از مشکلاتی مانند vanishing gradient و exploding gradient جلوگیری کنند و فرآیند آموزش به‌طور مؤثر و سریع‌تر پیش برود. مقداردهی اولیه نادرست می‌تواند منجر به کندی در همگرایی یا حتی عدم همگرایی مدل شود. علت نیاز به مقداردهی اولیه این است که وزن‌ها باید به‌طور تصادفی شروع شوند تا شبکه بتواند الگوهای مختلف داده‌ها را یاد بگیرد و همگرایی به‌درستی اتفاق بیفتد. برای اینکار می‌توان از مقداردهی اولیه به صورت تصادفی یا با استفاده از توزیع گوسین استفاده کرد.

۳

ابتدا فروارد را به صورت زیر تعریف می‌کنیم.

$$z_1 = w_1^T.X + b_1$$
$$a_1 = sigmoid(z_1)$$
$$z_2 = w_2^T.a_1 + b_2$$
$$a_2 = sigmoid(z_2)$$

حال بکوارد را به شکل زیر تعریف می‌کنیم.

$$dz_2 = a_2 - y,$$
$$db_2 = \frac{1}{m}\sum dz,$$
$$dz_1 = da_1 * a_1 * (1 - a_1),$$
$$db1 = \frac{1}{m}\sum dz_1 \text{(on each row)}$$
$$dw_2 = \frac{1}{m}a_1.dz_2^T + \frac{\lambda}{m}w_2$$
$$da_1 = w_2.dz_2$$
$$dw_1 = \frac{1}{m}X.dz_1^T + \frac{\lambda}{m}w_1$$

حال آن را آموزش می‌دهیم.
هزینه اولیه (قبل از آموزش): 0.7009893919459316
ایپاک اول:

$$w_1 = \begin{bmatrix} 0.19847626 & -0.2978312 \\ 0.39933369 & 0.10018809 \end{bmatrix}$$
$$b_1 = \begin{bmatrix} 0.09861572 \\ -0.19788227 \end{bmatrix}$$
$$w_2 = \begin{bmatrix} 0.28883918 \\ -0.5068092 \end{bmatrix}$$
$$b_2 = \begin{bmatrix} 0.03160075 \end{bmatrix}$$

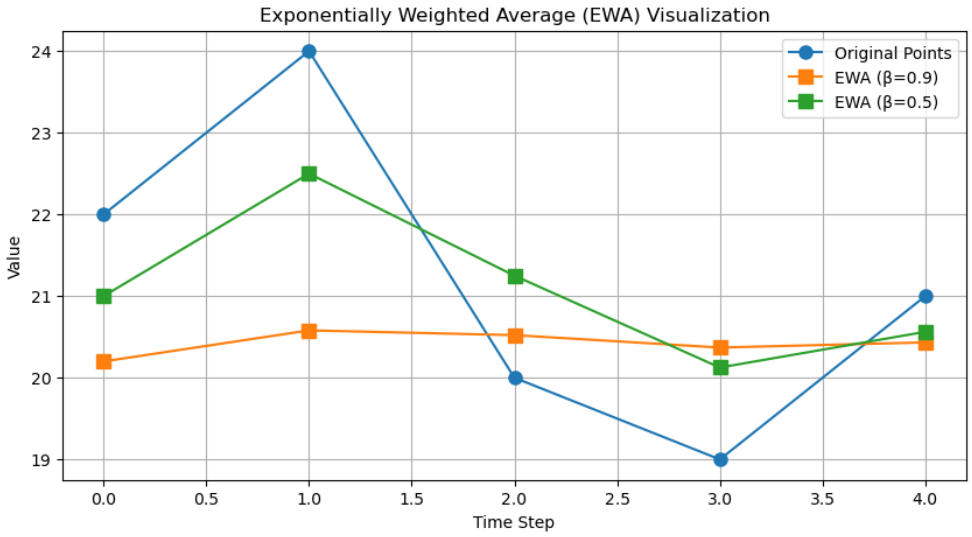
هزینه: 0.6958673142841637
ایپاک دوم:

$$w_1 = \begin{bmatrix} 0.19706267 & -0.29574618 \\ 0.39874567 & 0.10025999 \end{bmatrix}$$
$$b_1 = \begin{bmatrix} 0.09732586 \\ -0.19582172 \end{bmatrix}$$
$$w_2 = \begin{bmatrix} 0.2782244 \\ -0.51334582 \end{bmatrix}$$
$$b_2 = \begin{bmatrix} 0.01394589 \end{bmatrix}$$

هزینه: 0.6911723976554675
کد های پیاده سازی این بخش نیز در داخل این فایل موجود می‌باشد.

۴

EWA with beta 0.9: [20.2, 20.58, 20.522, 20.369799999999998, 20.43282]
EWA with beta 0.5: [21.0, 22.5, 21.25, 20.125, 20.5625]



شکل ۱: بررسی تاثیر مقدار بتا

همانطور که مشاهده می‌شود، اگر بتا زیاد باشد نوسانات و نویز داده ها کمتر میشود و الگوی صافی ایجاد میشود ولی وقتی بتا کمتر باشد، EWA به داده های لحظه ای نزدیک تر می‌شود و نویز آن بیشتر می‌شود.

فایل پیاده سازی