پوریا یزدانی ۴۰۰۲۴۳۰۸۲

تمرین چهارم - بخش تئوری

سوال اول)

الف)

ناپدید شدن گرادیان‌ها در بیشتر شبکه‌های عمیق و در هنگام backpropagation رخ می‌دهد. می‌دانیم در این شکبه‌ها تعداد لایه‌ها زیاد است بنابراین در تعداد مشتق‌هایی که نسبت به وزن‌های اولیه شبکه گرفته می‌شود بیشتر است.(طبق قاعده زنجیره‌ای) همچنین به خصوص اگر در این شبکه‌ها از فعال‌کننده‌هایی نظیر tanh و sigmoid استفاده شده باشد و خروجی لایه‌ها به بازه‌ای محدود شود این اعداد کوچک به مراتب در هم ضرب می‌شوند در نتیجه گرادیان‌های لایه‌های اولیه بسیار کوچک (ناپدید) می‌شوند و در نتیجه در زمان آموزش وزن‌های این لایه‌ها نمی‌توانند به خوبی بروزرسانی شوند.

از طرفی CNN ها به طور کلی شبکه‌های عمیق‌تر نسبت به MLP ها حساب می‌شوند زیرا در آن‌ها لایه‌های feature extraction نیز داریم که در آن‌ها تعدادی زیادی لایه‌های پیچشی روی هم استک شده‌اند. لایه‌های اولیه پیچشی معمولا مسئول extractکردن ویژگی‌های سطح پایین‌تر و ساده تر مانند گوشه‌ها و edgeها است. در نتیجه شبکه نمی‌تواند این ویژگی‌ها پایه‌ای را به خوبی شناسایی و یاد بگیرد و این ناتوانی در دیگر قسمت‌های شبکه نیز به صورت دومینویی ادامه پیدا می‌کند چون ویژگی‌های پایه‌ای را نتوانسته یاد بگیرد. همچنین به طور کلی اگر شبکه بسیار عمیق باشد ممکن است داده در روند پیمایش لایه‌ها معنی خودش را از دست بدهد و از هر لایه به لایه بعدی ممکن است اطلاعات بیشتری از دست برود که این خود باعث performance degregation است.

ب)

اگر فرض کنیم در یک لایه پیچشی مقداردهی اولیه فیلترها (کرنل) همگی ۰ و یا به طور کلی یک مقدار ثابت c باشد با توجه به اینکه ورودی یکسان است و وزن‌های یکسان (همان درایه‌های ماتریس فیلتر) خروجی یکسان می‌دهند و در نتیجه گرادیان‌های آن نیز در فرایند آموزش مثل هم می‌شوند و بروزرسانی وزن‌ها متقارن (مثل هم) می‌شوند در نتیجه شبکه representation

ها و ویژگی‌های جدیدی یاد نمیگیرد و واحدها کانولوشنی در این لایه (به جز یکی) اضافی می‌شوند و این تقارن در طول آموزش شکسته نمی‌شود.

مقدار دهی Xavier برای توابع فعال‌ساز sigmoid و tanh طراحی شده و مقادیر اولیه با توجه به یک توزیع (معمولا گوسی) و با واریانس مشخص انتخاب می‌شوند. هدف این است که واریانس گرادیان‌ها در لایه‌های مختلف حفظ شود. واریانس نیز با فرمول پایین تعیین می‌شود.

$$var = 1/(n_{in} + n_{out})$$

تعداد نورون‌های ورودی و خروجی آن لایه به ترتیب n_{in} و n_{out} می‌باشد.

در مقداردهی He که به صورت ویژه برای Relu و LeakyRelu و شاخه‌های آن طراحی شده مانند Xavier از یک توزیع مانند یونیفرم یا گوسی استفاده می‌شود و تفاوت آن در واریانس توزیع است که صرفا در آن n_{in} در نظر گرفته می‌شود:

$$var = 2/n_{in}$$

برای تثبیت فرایند آموزش و دوری از performance degregation نیاز به این داریم باید واریانس activation ها را در لایه‌های مختلف کنترل‌شده نگه داریم تا گرادیان ها دچار vanish و یا حتی explode نشوند. از آنجایی که در ReLu تقریبا ۵۰ درصد واریانس از بین می‌رود زیرا مقادیر منفی ۰ می‌شوند. ما سعی می‌کنیم این واریانس از دست رفته را با ۲ در صورت جبران کنیم (واریانس بیشتر) و همینطور نبود n_{out} نیز برای این است که بیشتر واریانس در frw pass رخ می‌دهد و رفتار variance propagation در لایه‌ها بیشتر توسط n_{in} تعیین می‌شود.

در نتیجه هر لایه تقریباً واریانسی یکسان را حفظ می‌کند و این امر توزیع‌های پایدار فعال‌سازی را در سراسر شبکه تضمین می‌کند. واریانس مناسب در فعال‌سازی به این معناست که گرادیان‌ها به‌خوبی توزیع می‌شوند، که این امر باعث می‌شود آموزش سریع‌تر و همگرایی روان‌تر انجام شود.

ج)

همانطور که گفته شد دو مشکل اصلی در شبکه‌های عمیق، ناپدید شدن گرادیان‌ها و از دست رفتن اطلاعات مهم و تضعیف آن در رفتن از لایه به لایه دیگر است. بلوک‌های

residual با دادن ورودی یک لایه به خروجی‌اش به صورت مستقیم یک یا چند لایه را رد (بایپس) می‌کنند.

$$y = F(x) + x$$

اتصالات میان‌بر به گرادیان‌ها اجازه می‌دهند تا لایه‌های میانی را دور بزنند و مستقیماً از طریق مسیر باقی‌مانده به عقب جریان پیدا کنند.این کار **مشکل ناپدید شدن گرادیان** را کاهش می‌دهد، زیرا گرادیان‌ها می‌توانند آزادانه‌تر به لایه‌های اولیه منتقل شوند.

با افزودن **ورودی** مستقیماً به **خروجی** شبکه می‌تواند در صورت نیاز یک **تابع همانی** را یاد بگیرد.این امر بهینه‌سازی را آسان‌تر می‌کند، زیرا شبکه نیازی به یادگیری تبدیلات غیرضروری ندارد.همانطور که ذکر شد مشکل از بین رفتن اطلاعات هم با این تکنیک address می‌شود.**سیگنال ورودی اصلی** حفظ شده و به **خروجی تبدیل‌شده** اضافه می‌شود.

به طور کلی این اتصالات آموزش شبکه‌های بسیار عمیق را پایدار می‌کنند و به شبکه‌هایی با صدها لایه (مانند ResNet-152) اجازه می‌دهند به‌طور مؤثری همگرا شوند.

بهبود جریان در backpropagation :

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} \cdot \left(\frac{\partial F(x)}{\partial x} + 1 \right)$$

این 1+ باعث می‌شود در با مشکل ناپدید شدن گرادیان مقابله کنیم.(اثر دادن ورودی مستقیم به خروجی)

سوال دوم)

Subject: Date: / /

سوال ۲)

convolution Layer
- Filter (kernel) size : ۳×۳
- num filters : ۱
- padding : valid → no padding
- stride : ۱

$$\text{output dimension} = \left\lfloor \frac{n_{in} - k + 2p}{s} \right\rfloor + 1$$

(k: kernel size, p: padding (0), s: stride)

$$\Rightarrow n_{out} = \left\lfloor \frac{۴ - ۳ + ۰}{۱} \right\rfloor + ۱ = ۲$$

$$X = \begin{bmatrix} ۱ & ۲ & ۱ & ۰ \\ ۲ & ۱ & ۰ & ۱ \\ ۱ & ۰ & ۲ & ۱ \\ ۰ & ۱ & ۲ & ۱ \end{bmatrix} \qquad W = \begin{bmatrix} ۱ & -۱ & ۰ \\ ۰ & ۱ & -۱ \\ -۱ & ۰ & ۱ \end{bmatrix} \qquad \frac{\partial L}{\partial Y} = \begin{bmatrix} ۱ & -۱ \\ -۱ & ۱ \end{bmatrix}$$

الف)

$$X = \begin{bmatrix} x_{۱۱} & x_{۱۲} & x_{۱۳} & x_{۱۴} \\ x_{۲۱} & x_{۲۲} & x_{۲۳} & x_{۲۴} \\ x_{۳۱} & x_{۳۲} & x_{۳۳} & x_{۳۴} \\ x_{۴۱} & x_{۴۲} & x_{۴۳} & x_{۴۴} \end{bmatrix} \qquad W = \begin{bmatrix} w_{۱۱} & w_{۱۲} & w_{۱۳} \\ w_{۲۱} & w_{۲۲} & w_{۲۳} \\ w_{۳۱} & w_{۳۲} & w_{۳۳} \end{bmatrix}$$

$$Y = \begin{bmatrix} y_{۱۱} & y_{۱۲} \\ y_{۲۱} & y_{۲۲} \end{bmatrix} \rightsquigarrow \begin{cases} y_{۱۱} = \sum_{m=۰}^{۳} \sum_{n=۰}^{۳} X[۱+m^{-1}, ۱+n^{-1}] \cdot w[m,n] \\ y_{۱۲} = \sum_{m=۰}^{۳} \sum_{n=۰}^{۳} X[۱+m^{-1}, ۲+n^{-1}] \cdot w[m,n] \\ y_{۲۱} = \sum_{m=۰}^{۳} \sum_{n=۰}^{۳} X[۲+m^{-1}, ۱+n^{-1}] \cdot w[m,n] \\ y_{۲۲} = \sum_{m=۰}^{۳} \sum_{n=۰}^{۳} X[۲+m^{-1}, ۲+n^{-1}] \cdot w[m,n] \end{cases}$$

$$Y[i,j] = \sum_{m=۱}^{H_W} \sum_{n=۱}^{W_W} X[i+m, j+n] w[m,n]$$

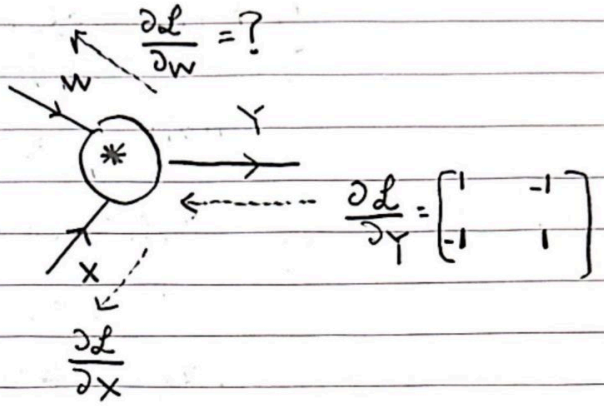
For stride=1 and no padding

$$\rightsquigarrow \begin{cases} y_{۱۱} = ۱+(-۲)+۰+۰+۱+۰+۰+(-۱)+۰ = -۱ \\ y_{۱۲} = ۲+(-۱)+۰+۰+۰+(-۱)+۰+۰+۱ = ۱ \\ y_{۲۱} = ۲+۰+۰+۰+۲+(-۱)+(-۱)+۰+۱ = ۲ \\ y_{۲۲} = ۱+۰+۰+۰+۰+۲+(-۱)+(-۱)+۱ = ۲ \end{cases}$$

Subject: Date: / /

$\Rightarrow Y = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$ end of forward pass

backward pass:



we have this

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{\partial \mathcal{L}}{\partial Y} \frac{\partial Y}{\partial W}$$ → we need this → $\frac{\partial Y[i,j]}{\partial W[m,n]} = X[i+m, j+n]$

↳ $\frac{\partial Y}{\partial W[m,n]} = \sum_i \sum_j X[i+m, j+n]$

$$\Rightarrow \frac{\partial \mathcal{L}}{\partial W[m,n]} = \sum_i \sum_j \frac{\partial \mathcal{L}}{\partial Y[i,j]} \frac{\partial Y[i,j]}{\partial W[m,n]}$$

$$= \sum_i \sum_j \frac{\partial \mathcal{L}}{\partial Y[i,j]} X[i+m-1, j+n-1]$$

$$\Rightarrow \frac{\partial \mathcal{L}}{\partial W[1,1]} = \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{\partial \mathcal{L}}{\partial Y[i,j]} \cdot X[i+1-1, j+1-1] = 1 + (-2) + (-2) + i = -2$$

$$\frac{\partial \mathcal{L}}{\partial W[1,2]} = \sim \sim X[i+1-1, j+2-1] = 2 + (-1) + (-1) + 0 = 0$$

$$\frac{\partial \mathcal{L}}{\partial W[2,1]} = \sim \sim X[i+1-1, j+2-1] = 1 + 0 + 0 + 1 = 2$$

$$\frac{\partial \mathcal{L}}{\partial W[2,2]} = \sim \sim X[i+2-1, j+1-1] = 2 + (-1) + (-1) + 0 = 0$$

$$\frac{\partial \mathcal{L}}{\partial W[2,2]} = \sim \sim X[i+2-1, j+2-1] = 1 + 0 + 0 + 2 = 4$$

s.a.m

$$\frac{\partial \mathcal{L}}{\partial w[2,2]} = \sum\sum \frac{\partial \mathcal{L}}{\partial Y} \cdot X[i+2-1, j+2-1] = 0+(-1)+(-2)+1 = -2$$

$$\frac{\partial \mathcal{L}}{\partial w[2,1]} = \sum\sum \frac{\partial \mathcal{L}}{\partial Y} \cdot X[i+2-1, j+1-1] = 1+0+0+1 = 2$$

$$\frac{\partial \mathcal{L}}{\partial w[1,2]} = \sum\sum \frac{\partial \mathcal{L}}{\partial Y} \cdot X[i+1-1, j+2-1] = 0+(-2)+(-1)+0 = -3$$

$$\frac{\partial \mathcal{L}}{\partial w[1,1]} = \sum\sum \frac{\partial \mathcal{L}}{\partial Y} \cdot X[i+1-1, j+1-1] = 2+(-1)+(-2)+1 = 0$$

$$\Rightarrow \frac{\partial \mathcal{L}}{\partial w} = \begin{bmatrix} 0 & -3 \\ 2 & -2 \end{bmatrix}$$

we have this

$$\frac{\partial \mathcal{L}}{\partial X} = \frac{\partial \mathcal{L}}{\partial Y} \cdot \frac{\partial Y}{\partial X} \rightarrow \text{we need this}$$

$$\frac{\partial \mathcal{L}}{\partial x_i} = \sum_k \frac{\partial \mathcal{L}}{\partial y_k} \frac{\partial y_k}{\partial x_i}$$

$$Y[1,1] = \sum_m \sum_n X[1+m-1, 1+n-1] \cdot w[m,n]$$

$$\vdots$$

$$\vdots \rightarrow \text{فرمول صفحه اول}$$

بعنوان مثال $X[1,1]$ فقط در محاسبه $Y[1,1]$ ظاهر می‌شود. پس

$$\frac{\partial \mathcal{L}}{\partial X[1,1]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot \frac{\partial Y[1,1]}{\partial X[1,1]}$$

$$= \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot w[1,1] = 1 \cdot 1 = 1$$

* یعنی باید ببینیم که $X[i,j]$ در کدام $Y[i,j]$ ها contribution داشته و $\frac{\partial \mathcal{L}}{\partial Y[i,j]}$ متناظر با آن‌ها را جمع می‌زنیم.

contributes
→

X[1,1] → Y[1,1]
X[1,2] → Y[1,0], Y[1,1]
X[1,3] → Y[1,1], Y[1,2]
X[1,4] → Y[1,2]

X[2,1] → Y[1,1], Y[2,1]
X[2,2] → Y[1,1], Y[1,2], Y[2,1], Y[2,2]
X[2,3] → Y[1,1], Y[1,2], Y[2,1], Y[2,2]
X[2,4] → Y[1,2], Y[2,2]

X[3,1] → Y[1,1], Y[2,1]
X[3,2] → Y[1,1], Y[1,2], Y[2,1], Y[2,2]
X[3,3] → Y[1,1], Y[1,2], Y[2,1], Y[2,2]
X[3,4] → Y[1,2], Y[2,2]

X[4,1] → Y[2,1]
X[4,2] → Y[2,1], Y[2,2]
X[4,3] → Y[2,1], Y[2,2]
X[4,4] → Y[2,2]

$\frac{\partial \mathcal{L}}{\partial Y_k}$ ✱ در هر $\frac{\partial \mathcal{L}}{\partial X[i,j]}$ برای آن Yهایی که وابسته ظاهر شده در X را در نظر می‌گیریم.

$$\frac{\partial \mathcal{L}}{\partial X[i,j]} = \sum_k \frac{\partial \mathcal{L}}{\partial y_k} \frac{\partial y_k}{\partial X[i,j]} \quad \text{for } k \text{ in above } (a,b)$$

$$\frac{\partial \mathcal{L}}{\partial X[1,2]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot w[1,2] + \frac{\partial \mathcal{L}}{\partial Y[1,2]} \cdot w[1,1] = (-1) + (-1) = -2$$

$$\frac{\partial \mathcal{L}}{\partial X[1,3]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot w[1,3] + \frac{\partial \mathcal{L}}{\partial Y[1,2]} \cdot w[1,2] = 0 + 1 = 1$$

$$\frac{\partial \mathcal{L}}{\partial X[2,4]} = \frac{\partial \mathcal{L}}{\partial Y[1,2]} \cdot w[2,1] = 0$$

$$\frac{\partial \mathcal{L}}{\partial X[2,1]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot w[2,1] + \frac{\partial \mathcal{L}}{\partial Y[2,1]} \cdot w[1,1] = 0 + (-1) = -1$$

$$\frac{\partial \mathcal{L}}{\partial X[2,2]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} \cdot w[2,2] + \frac{\partial \mathcal{L}}{\partial Y[1,2]} \cdot w[1,2] + \frac{\partial \mathcal{L}}{\partial Y[2,1]} \cdot w[1,1] + \frac{\partial \mathcal{L}}{\partial Y[2,2]} \cdot w_{11} = 1+0+1+1 = 3$$

$$\frac{\partial \mathcal{L}}{\partial X[2,3]} = \frac{\partial \mathcal{L}}{\partial Y[1,1]} w[2,3] + \frac{\partial \mathcal{L}}{\partial Y[1,2]} w[2,2] + \frac{\partial \mathcal{L}}{\partial Y[2,1]} w[1,3] + \frac{\partial \mathcal{L}}{\partial Y[2,2]} w[1,2] = -3$$

$$\frac{\partial \mathcal{L}}{\partial [3,2]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[2,1]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[2,2]^{(2)} = 1 \cdot (-2) = -2$$

$$\frac{\partial \mathcal{L}}{\partial [3,1]A^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[3,1]^{(2)} + \frac{\partial \mathcal{L}}{\partial [1,2]A^{(2)}} \cdot w[3,2]^{(2)} = 1 \cdot (-1) = -1$$

$$\frac{\partial \mathcal{L}}{\partial [2,3]A^{(1)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[1,3]^{(2)} + \frac{\partial \mathcal{L}}{\partial [1,2]A^{(2)}} \cdot w[2,3]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[2,2]^{(2)} + \frac{\partial \mathcal{L}}{\partial [1,2]A^{(2)}} \cdot w[2,1]^{(2)} = 0$$

$$\frac{\partial \mathcal{L}}{\partial [2,3]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[3,3]^{(2)} + \frac{\partial \mathcal{L}}{\partial [1,2]A^{(2)}} \cdot w[3,2]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,1]A^{(2)}} \cdot w[2,1]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[2,2]^{(2)} = -1$$

$$\frac{\partial \mathcal{L}}{\partial [3,3]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[3,3]^{(2)} = 1$$

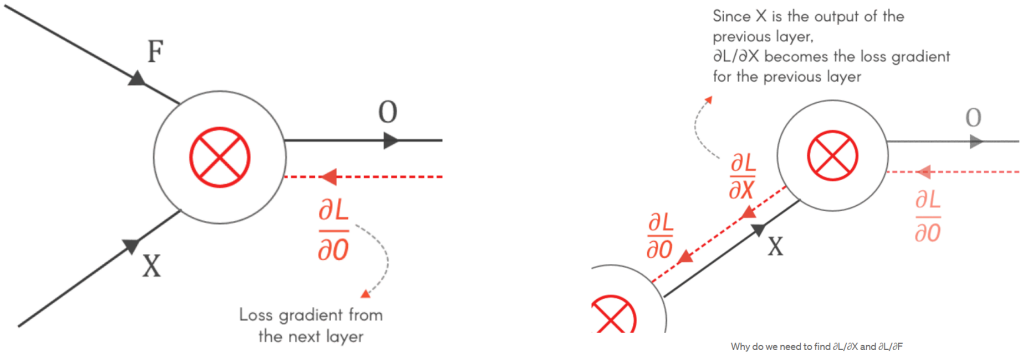
$$\frac{\partial \mathcal{L}}{\partial [3,4]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [2,1]A^{(2)}} \cdot w[4,1]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[4,2]^{(2)} = -2$$

$$\frac{\partial \mathcal{L}}{\partial [2,3]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[3,3]^{(2)} + \frac{\partial \mathcal{L}}{\partial [1,2]A^{(2)}} \cdot w[3,2]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,1]A^{(2)}} \cdot w[2,1]^{(2)} + \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[2,2]^{(2)} = -1$$

$$\frac{\partial \mathcal{L}}{\partial [2,4]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [1,1]A^{(2)}} \cdot w[4,3]^{(2)} = 1$$

$$\frac{\partial \mathcal{L}}{\partial [3,4]X^{(2)}} = \frac{\partial \mathcal{L}}{\partial [2,2]A^{(2)}} \cdot w[4,3]^{(2)} = 1$$

$$\Longrightarrow \frac{\partial \mathcal{L}}{\partial X^{(2)}} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -2 & 3 & 0 & -1 \\ -1 & 3 & -2 & 1 \\ 1 & -2 & 1 & 0 \end{bmatrix}$$



همینطور که در شکل‌های بالا می‌بینیم، می‌دانیم در انتشار رو به جلو خروجی هر لایه کانولوشنی ورودی لایه بعدی‌اش می‌باشد. اگر هر کدام از این لایه‌ها را به عنوان یک بلوک با ۳ عنصر ورودی، فیلتر و خروجی نگاه کنیم. به ما کمک می‌کند تا ببینیم که در انتشار رو به عقب ورودی هر بلوک، گرادیان خطای لایه بعدی نسبت به خروجی‌اش است که این خروجی با توجه به ورودی‌اش حساب شده ($conv(X,W)$) و از طرفی ورودی لایه بعد درواقع همان **خروجی** لایه کنونی است. پس درواقع از مقدار گرادیان خطای لایه بعد نسبت به ورودی‌اش(خروجی لایه کنونی) می‌توان به عنوان **گرادیان خطا نسبت به خروجی این لایه** نگاه کرد.

در شبکه‌های عصبی کانولوشنی، گرادیان تابع خطا نسبت به ورودی هر لایه نشان می‌دهد که هر پیکسل یا ویژگی در ورودی چقدر در خطای نهایی تاثیر داشته است. این گرادیان $\frac{\partial L}{\partial X}$ به‌عنوان **سیگنال خطا** به لایه‌های قبلی منتقل می‌شود. به‌عبارت دیگر، خروجی هر لایه به‌عنوان ورودی لایه قبلی در نظر گرفته شده و این روند به‌صورت بازگشتی ادامه می‌یابد تا وزن‌های تمام لایه‌ها برای کاهش خطای نهایی به‌روزرسانی شوند.

سوال سوم)

برای استفاده از RNN ابتدا مانند هر وظیفه NLP دیگر باید ورودی و درواقع اینجا کلمات را تبدیل به بردارهای عددی کنیم. مثلا می‌توانیم از الگوریتم word2vec استفاده کنیم. این دنباله از امبدینگ‌ها کلمه به کلمه وارد شبکه می‌شود. در هر گام شبکه امبدینگ کلمه کنونی و خروجی لایه hidden قبلی را دریافت می‌کند و خروجی لایه مخفی کنونی را با توجه به فرمول پایین محاسبه می‌کند:

$$h_t = f(W_h \cdot w_t + U_h \cdot h_{t-1} + b_h)$$

بعد از پردازش تمام n کلمه ورودی خروجی لایه مخفی آخر به عنوان نمایشی از کل جمله عمل می‌کند. این استیت آخر contextual meaning و احساسات بیان شده در جمله را در بر دارد و در ادامه این نمایش معنادار از جمله وارد یک لایه fully connected می‌شود و در نهایت یک لایه softmax برای پیشبینی مقادیر بین ۰ تا ۴ همینطور از تابع خطا CE استفاده می‌کنیم.
به طور خاص در زمان inference (آزمایش) pipeline پایین اجرا می‌شود.

- کلمات جمله ورودی به از طریق یک الگوریتم امبدینگ به بردارهای عددی تبدیل می‌شوند.
- شبکه هر کلمه را به صورت sequential و با نگه داشتن حالت مخفی در هر گام زمانی پردازش می‌کند.
- خروجی لایه آخر مخفی final hidden state است که نمایشی معنادار از جمله ورودی است.
- این استیت مخفی پایانی وارد یک لایه fully connected می‌شود با فعالساز softmax می‌شود می‌شود.
- کلاس با بیشترین امتیاز به عنوان لیبل آن جمله خروجی داده می‌شود.

البته حالت‌های دیگری نیز از embedding وجود دارد مانند BPE یا WordPiece. این‌ها الگوریتم‌های subword tokenization هستند و هر کلمه می‌تواند تبدیل به چند بردار شود. اما دیگر مراحل ثابت است.
الف)
با توجه به تکنیک‌هایی که وجود دارد دو حالت را می‌توان برای تعداد استفاده شدن از softmax متصور شد. یک حالت این است که صرفا در آخر مانند روندی که توصیف شد ۱ بار softmax را صدا بزنیم و کلاس با بیشترین امتیاز را اعلام کنیم. راه دیگر این است که در هر a گام زمانی خروجی مخفی آن لایه را به softmax (پس از گذشتن از لایه fully connected) بدهیم یا هر بار که کلمه خاص یا علامت نگارشی خاص(conditional). در این روش پس از پردازش تمام کلمات ما تعدادی خروجی softmax داریم که هرکدام sentimentآن جمله را از اول تا جای مشخصی در بر دارد. ما می‌توانیم از روش‌های مختلف مانند میانگین‌گیری برای aggregate کردن این خروجی‌ها استفاده کنیم و سپس کلاس با بیشترین امتیاز را اعلام کنیم.

- در روش اول ۱ بار softmax صدا می‌شود.

- در روش دوم بین ۱ تا n بار softmax می‌تواند صدا شود.(این روش برای جملات طولانی بهتر است زیرا کلماتی که معنای جمله را دربر دارند به صورت یونیفرم در جمله پخش نشدند و همینطور با ناپدید شدن گرادیان نیز می‌توان مقابله کرد.)

در نظر داریم اگر از الگوریتم‌های subword tokenization استفاده کنیم به ازای جمله n کلمه‌ای می‌توانیم بیش از n بردار embedding داشته باشیم و در نتیجه این امکان وجود دارد در این استراتژی بیش از n بار softmax صدا شود.

ب)

هر خروجی softmax یا همان $\widehat{y_n}$ در شبکه موردنظر برای تحلیل احساسات، یک **توزیع احتمالاتی روی کلاس‌های احساسات** (مثلاً منفی، خنثی، مثبت) را نشان می‌دهد. این توزیع نشان می‌دهد که جمله (یا کلمه، بسته به استراتژی خروجی) با چه احتمالی به هر کلاس تعلق دارد. مقادیر این توزیع از طریق یک لایه کاملاً متصل و سپس یک تابع **Softmax** به دست می‌آید. در نهایت، کلاسی که بالاترین احتمال را دارد، به‌عنوان **برچسب نهایی احساسات** پیش‌بینی می‌شود. بسته به استراتژی استفاده‌شده (خروجی در زمان نهایی، در هر گام، یا به‌صورت شرطی)، این توزیع می‌تواند در یک یا چند مرحله تولید شود.

پس درواقع هر خروجی softmax نشان‌دهنده یک توزیع احتمالی روی کلاس‌های احساسات می‌باشد که می‌توان به صورت زیر نشان داد:

$$\widehat{y_n} = [P(c_0|x), P(c_1|x), ..., P(c_4|x)]$$

ج)

اگر منظور از خروجی مدل در هر گام خروجی، خروجی لایه‌های مخفی باشد، همانطور که توضیح داده شد ورودی‌ها در هر گام زمانی t شامل امبدینگ کلمه کنونی یعنی x_t و خروجی لایه مخفی قبل یعنی h_{t-1} می‌باشد که شامل معنای جمله تا کلمه t ام است.

اگر منظور از خروجی در هر گام زمانی خروجی softmax است، خروجی آن لایه مخفی وارد یک لایه تماما متصل می‌شود و سپس به عنوان ورودی به softmax داده می‌شود.

سوال چهارم)

الف)

با توجه به مقاله لایه پیچشی آخر در نظر گرفته می‌شود. در این فرمول y_c امتیاز کلاس پیش‌بینی شده برای نمونه مورد نظر است.(قبل از اعمال softmax) پس درواقع امتیاز هر کانال لایه پیچشی مورد نظر حاصل global-average pooling روی گرادیان y_c نسبت به هر درایه k feature map ام بعد از activation است. H و W نیز ابعاد این feature map را نشان می‌دهند.

برای تفسیر این فرمول در نظر می‌گیریم که گرادیان حساب شده در این فرمول نشان‌دهنده این است که چقدر تغییر در A_{ij}^k تغییر در امتیاز کلاس انتخاب شده برای آن نمونه می‌شود.(y_c) اگر این گرادیان زیاد بود پس تاثیرپذیری زیادی دارد و برعکس. همینطور میانگین‌گیری روی کل feature map باعث می‌شود تا میزان تاثیر و تغییرات محاسبه‌شده محدود به برخی نقاط خاص آن feature map نباشد و نویزها را کمتر کنیم و امتیاز نهایی محاسبه شده برای وزن آن کانال را با توجه به کل آن محاسبه کنیم. همچنین با تمرکز صرفا بر روی logit کلاس پیش‌بینی شده مطمئن می‌شویم تنها نواحی مربوط به کلاس c شناسایی می‌شوند. پس به طور کلی این خاصیت گرادیان که میزان حساسیت خروجی نهایی به مقادیر میانی را اندازه‌گیری می‌کند پایه و اساس این فرمول و روش است.

ب)

می‌دانیم فعالساز ReLU تنها مقادیر مثبت را نگه می‌دارد و تمامی مقادیر منفی feature map را با ۰ جایگزین می‌کند از طرفی به طول کلی در میانگین وزندار feature map ها که وزن‌های آن با توجه به کلاس پیش‌بینی شده c محاسبه شده مقادیر مثبت نشان‌دهنده positive contribution به کلاس c می‌باشد زیرا این پیکسل‌ها بیشترین تاثیر را روی کلاس c داشته‌اند و برعکس یعنی مقادیر منفی contributionای به این کلاس نداشته‌اند و احتمالا contribution آن‌ها برای کلاس‌های دیگر بوده است و یا مربوط به ویژگی‌هایی است که برای تشخیص این کلاس مهم نبوده‌اند.(ذکر شده در قسمت 3 مقاله). اگر این ReLU را اعمال نکنیم برخی اوقات هیت مپ بدست آمده نواحی مربوط به کلاس‌های دیگر را هایلایت می‌کند. همینطور این حذف این مقادیر باعث می‌شود هیت مپ نهایی واضح‌تر و discriminative تر باشد و نویزها را در نظر نگیرد.