دانشکده مهندسی کامپیوتر

تمرین سری چهارم

درس پردازش زبان های طبیعی

پوریا رحیمی - 99521289

1403-1404

سوال اول) دو جدول زیر تعداد Unigram ها و Bigram ها در یک پیکره فرضی را نشان می دهد.

ما	خواندیم	دیروز	امروز	داستان	کتاب
1872	1495	2021	1943	945	1245

	ما	خواندیم	دیروز	امروز	داستان	کتاب
ما	0	156	411	452	238	387
خواندیم	2	0	6	11	84	112
دیروز	341	32	0	48	68	254
امروز	328	12	0	0	87	231
داستان	4	345	84	38	0	1
کتاب	31	320	3	0	403	0

احتمال رخداد جملات زیر را محاسبه کنید. فرض بر این است که جملات تست در وسط یک رشته هستند. یعنی در نظر گرفتن احتمال بند های شروع و پایان جمله لازم نیست.

جمله تست 1 : ... ما امروز کتاب خواندیم ...

جمله تست 2 : ... ما دیروز داستان خواندیم ...

برای محاسبه احتمال رخداد جملات تست، ابتدا احتمال رخداد هر کلمه را در جایگاه مورد نظر در نظر می‌گیریم (برای مثال برای جمله ما امروز ... با توجه به اینکه احتمال بند های شروع و پایان در نظر گرفته نمی شود می شود احتمال ما به شرط امروز و بقیه نیز به همین صورت پیش می رویم) و سپس این احتمالات را با یکدیگر ضرب می‌کنیم. احتمال رخداد هر کلمه را از جدول داده شده می‌خوانیم.

در نتیجه برای جمله تست اول داریم :

احتمال ما به شرط امروز = $\frac{452}{1872}$ = 0.24

احتمال امروز به شرط کتاب = $\frac{231}{1943}$ = 0.119

احتمال کتاب به شرط خواندیم = $\frac{320}{1245}$ = 0.257

احتمال جمله تست اول = 0.257 x 0.119 x 0.24 = 0.00734

برای جمله تست دوم داریم :

احتمال ما به شرط دیروز = $\frac{411}{1872}$ = 0.22

احتمال دیروز به شرط داستان = $\frac{68}{2021}$ = 0.033

احتمال داستان به شرط خواندیم = $\frac{345}{945}$ = 0.365

احتمال جمله تست دوم = 0.365 x 0.033 x 0.22 = 0.00265

سوال دوم) رابطه زیر را اثبات کنید.

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1}) = \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

این مسئله مربوط Markov chain می باشد ، این مسئله میخواهد بگویید که احتمال رویداد در یک زنجیره به رویداد قبلی وابسته است و حال از ما خواسته شده است که این را ثابت کنیم.
برای اثبات این قضیه ما از تعریف شرطی بودن احتمال استفاده می کنیم :

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

طبق فرمولی که در صفحه قبل دیدیم می توانیم آن را برای اثبات مسئله خودمان تعمیم دهیم
برای این کار می توان فرمول را به این صورت نوشت :

$$P(W_n|W1\,,W2,W3\,,\dots,Wn-1) = \frac{P(W1\,,W2\,,W3\,,\dots,Wn)}{P(W1\,,W2,W3\,,\dots,Wn-1)}$$

این فرمول را می توان به صورت ساده تری نیز نوشت :

$$P(W_n|W_1^{n-1}) = \frac{P(W_1^n)}{P(W_1^{n-1})}$$

با یک طرفین وسطین ساده داریم :

$$P(W_1^n) = P(W_n|W_1^{n-1}) * P(W_1^{n-1})$$

حالا احتمال $P(W_1^{n-1})$ را بسط می دهیم :

$$P(W_1^{n-1}) = P(W_{n-1}|W_1^{n-2}) * P(W_1^{n-2})$$
$$P(W_1^{n-2}) = P(W_{n-2}|W_1^{n-3}) * P(W_1^{n-3})$$

.

.

.

$$P(W_1^n) = P(W_n|W_1^{n-1}) * P(W_{n-1}|W_1^{n-2}) * P(W_{n-2}|W_1^{n-3}) * \dots P(W_2|W_1) * P(W_1)$$

حالا با توجه به فرمول بالا می توان این نتیجه را گرفت که بسط بالا برابر با فرمول زیر می باشند :

$$P(W_1^n) = \prod_{k=1}^{n} P(W_K|W_1^{K-1})$$

سوال سوم)

a) در مرحله زمانی 1، beam search کدام دنباله را ذخیره می کند؟

$$Log\big(P(neural \mid < START >)P(< START >)\big) = -65$$
$$Log\big(P(network \mid < START >)P(< START >)\big) = -73$$

سپس بعد از بدست آوردن احتمال هر شاخه آن ها را بر حسب فاصله های که بدست آورده ایم مرتب می کنیم (بر اساس بهینه ترین حالت مرتب می کنیم) بعد از آن *K* تا از آن ***sequences*** ها را که در سوال گفته شده است *K=2* انتخاب می کنیم و آن ها را ذخیره می کنیم. که در این جا دو موارد زیر انتخاب می شوند و ذخیره می شوند :

$$\{< START >, neural\} \rightarrow -65$$
$$\{< START >, network\} \rightarrow -73$$

b) در مرحله زمانی 2، beam search کدام دنباله را ذخیره می کند؟

$$Log\big(P(neural \mid < START >, neural)P(< START >, neural)\big) = -0.8 - 65 = -65.8$$

$$Log\big(P(neural \mid < START >, network)P(< START >, network)\big) = -0.6 - 73 = -73.6$$

$$Log\big(P(network \mid < START >, neural\ )P(< START >, neural)\big) = -0.6 - 65 = -65.6$$

$$Log\big(P(network \mid < START >, network)P(< START >, network)\big) = -0.8 - 73 = -73.8$$

سپس بعد از بدست آوردن احتمال هر شاخه آن ها را بر حسب فاصله های که بدست آورده ایم مرتب می کنیم (بر اساس بهینه ترین حالت مرتب می کنیم) بعد از آن *K* تا از

آن ***sequences*** ها را که در سوال گفته شده است ***K=2*** انتخاب می کنیم و آن ها را ذخیره می کنیم. که در این جا دو موارد زیر انتخاب می شوند و ذخیره می شوند :

$$\{< START >, neural, network\} \rightarrow -65.6$$
$$\{< START >, neural, neural\} \rightarrow -65.8$$

c) در مرحله زمانی 3، beam search کدام دنباله را ذخیره می کند؟

$$Log\big(P(neural \mid < START >, neural, network)P(< START >, neural, network)\big) = -0.8 - 65.6 = -66.4$$

$$Log\big(P(neural \mid < START >, neural, neural)P(< START >, neural, neural)\big) = -0.01 - 65.8 = -65.81$$

$$Log\big(P(network \mid < START >, neural, network)P(< START >, neural, network)\big) = -0.6 - 65.6 = -66.2$$

$$Log\big(P(network \mid < START >, neural, neural)P(< START >, neural, neural)\big) = -4.6 - 65.8 = -70.4$$

سپس بعد از بدست آوردن احتمال هر شاخه آن ها را بر حسب فاصله های که بدست آورده ایم مرتب می کنیم (بر اساس بهینه ترین حالت مرتب می کنیم) بعد از آن ***K*** تا از آن ***sequences*** ها را که در سوال گفته شده است ***K=2*** انتخاب می کنیم و آن ها را ذخیره می کنیم. که در این جا دو موارد زیر انتخاب می شوند و ذخیره می شوند :

$$\{< START >, neural, neural, neural\} \rightarrow -65.81$$
$$\{< START >, neural, network, network\} \rightarrow -66.2$$

d) آیا beam search دنباله ی کلی با بیشترین احتمال در این مثال را برمیگرداند؟ توضیح دهید.

الگوریتم beam search به نحوی مانند الگوریتم A* عمل می کند و گام هایش را بر اساس یک هزینه تقریبی برای رسیدن به هدف (heuristic) انتخاب می کند در نتیجه با توجه به این موضوع ممکن است که دنباله ای با بیشترین احتمال را به ما ندهد چرا که ممکن است در نود های ابتدایی ما بر اساس K که داریم مسیری(دنباله ای) را حذف کنیم که در ادامه cost بهتری برای ما داشته باشد (احتمال بیشتری داشته باشد) در نتیجه نمی توان با قاطعیت گفت که همیشه دنباله ای با بهترین هزینه (بیشترین احتمال) را برای ما برمیگرداند ولی در این مثال دو دنباله ای که بهترین هزینه (بیشترین احتمال) را دارند را به ما می دهد.

e) پیچیدگی زمان اجرا تولید یک دنباله با طول T با اندازه پرتو K با RNN چقدر است؟ بر حسب T و K و M پاسخ دهید.

روند اجرای الگوریتم beam search به این صورت می باشد که K دنباله را در یک بخش از حافظه نگه می دارد سپس باید احتمال تمام کلمات را برای همه این K دنباله محاسبه کند سپس بعد از بدست آوردن احتمال کلمات برای همه K ها نوبت به آن می رسد که آن ها را بر اساس بهینه ترین cost (بیشترین احتمال) مرتب کند تا بتوانیم K تا از بالاترین احتمالات را نگه داریم و آن ها را ذخیره کنیم و این کار را باید برای کل دنباله به طول T انجام دهیم در نتیجه پیچیدگی زمانی اجرای این الگوریتم برابر است با :

$$O(M \times K \times T \times \log(k))$$

سوال چهارم) به سوالات زیر پاسخ دهید.

a) اگر در LSTM فقط بخواهیم گیت forget را داشته باشیم و گیت های input و output را حذف کنیم، چه اتفاقی می افتد و خروجی چه تغییری می کند؟

در مدل LSTM ، سه گیت وجود دارد: گیت فراموشی forget gate ، گیت ورودی input gate و گیت خروجی output gate این سه گیت با هم کار می‌کنند تا اطلاعات را در حافظه بلند مدت long-term memory ذخیره و بازیابی کنند.

اگر فقط گیت فراموشی را نگه داریم و دو گیت دیگر را حذف کنیم، LSTM قادر به ذخیره اطلاعات جدید یا بازیابی اطلاعات از حافظه بلند مدت نخواهد بود. گیت فراموشی تنها می‌تواند تصمیم بگیرد که چه اطلاعاتی از حافظه بلند مدت حذف شوند، اما بدون گیت ورودی، اطلاعات جدیدی برای جایگزینی اطلاعات حذف شده وجود ندارد. همچنین، بدون گیت خروجی، ما نمی‌توانیم اطلاعات را از حافظه بلند مدت بازیابی کنیم.

بنابراین، اگر فقط گیت فراموشی را نگه داریم، LSTM به مرور زمان اطلاعات را از دست می‌دهد و نمی‌تواند اطلاعات جدیدی را ذخیره کند یا اطلاعات موجود را بازیابی کند. این می‌تواند منجر به کاهش شدید عملکرد شبکه شود. خروجی نیز تغییر خواهد کرد، زیرا LSTM دیگر قادر به حفظ یا بازیابی اطلاعات بلند مدت نیست. این می‌تواند منجر به از دست رفتن ارتباطات زمانی در داده‌ها شود، که یکی از ویژگی‌های کلیدی LSTM است.

b) اگر در یک LSTM مقدار گیت forget را به صفر تنظیم کنیم، چه اتفاقی می افتد و چطور این تغییر تاثیری بر روی توانایی شبکه در یادگیری و پیش بینی دارد؟

گیت فراموشی در LSTM کنترل می‌کند که چه اطلاعاتی از حافظه بلند مدت حذف شوند. اگر مقدار گیت فراموشی را به صفر تنظیم کنیم، این به این معنی است که هیچ اطلاعاتی از حافظه بلند مدت حذف نمی‌شود. در واقع، LSTM تمام اطلاعات را به طور دائمی حفظ می‌کند.

این تغییر می‌تواند تاثیرات مختلفی بر روی توانایی شبکه در یادگیری و پیش‌بینی داشته باشد. از یک سو، حفظ کردن تمام اطلاعات می‌تواند به LSTM اجازه دهد تا الگوهای زمانی بسیار طولانی را یاد بگیرد، چیزی که در بسیاری از مدل‌های دیگر RNN غیرممکن است. از سوی دیگر، این می‌تواند منجر به مشکلاتی شود. به عنوان مثال، اگر LSTM تمام اطلاعات را حفظ کند، ممکن است "سر و صدا" یا اطلاعات غیرمرتبط را نیز حفظ کند که می‌تواند باعث کاهش عملکرد شود.

علاوه بر این، اگر LSTM هیچ اطلاعاتی را فراموش نکند، ممکن است با مشکل "گرادیان‌های ناپایدار" مواجه شود. این مشکل زمانی رخ می‌دهد که گرادیان‌ها در طول زمان به سرعت رشد یا کاهش می‌کنند، که می‌تواند باعث شود شبکه عصبی عمیق به سختی یاد بگیرد. این مشکل به خصوص مهم است زمانی که با داده‌های زمانی طولانی کار می‌کنیم.

بنابراین، در حالی که تنظیم کردن مقدار گیت فراموشی به صفر می‌تواند در برخی موارد مفید باشد، اما ممکن است در برخی موارد دیگر باعث ایجاد مشکلات شود. بنابراین، بسته به مسئله و داده‌هایی که با آنها کار می‌کنیم، ممکن است نیاز باشد تا مقدار گیت فراموشی را به طور دقیق تنظیم کنیم.

c) توضیح دهید که چگونه افزایش تعداد لایه های LSTM در یک شبکه می تواند به کارایی و عملکرد شبکه کمک کند یا باعث افزایش پیچیدگی شود؟

افزایش تعداد لایه‌های LSTM در یک شبکه می‌تواند هم به بهبود عملکرد شبکه کمک کند و هم باعث افزایش پیچیدگی شود. این دو جنبه را بررسی می کنیم:

بهبود عملکرد:

- **تعمیق یادگیری :** با افزایش تعداد لایه‌های LSTM ، شبکه قادر خواهد بود پیچیدگی‌های بیشتری را در داده‌ها کشف کند. این می‌تواند به شبکه کمک کند تا الگوهای پیچیده‌تر و در سطوح بالاتر را یاد بگیرد.

- **مدل‌سازی وابستگی‌های طولانی مدتLSTM :** ها به خاطر طراحی خاص خود، قادرند وابستگی‌های زمانی طولانی‌مدت را مدل کنند. با افزایش تعداد لایه‌ها، این قابلیت بیشتر می‌شود.

افزایش پیچیدگی:

- **پیچیدگی محاسباتی :** با افزایش تعداد لایه‌های LSTM ، تعداد پارامترهایی که باید آموخته شوند، افزایش می‌یابد. این می‌تواند منجر به افزایش زمان آموزش و نیاز به منابع بیشتر شود.
- **(Overfitting) :** اگر داده‌های کافی برای آموزش مدل وجود نداشته باشد، افزایش تعداد لایه‌ها می‌تواند منجر به بیش‌برازش شود. در این حالت، مدل به جای یادگیری الگوهای کلی، صرفاً داده‌های آموزشی را حفظ می‌کند که این موضوع باعث کاهش عملکرد در داده‌های تست می‌شود.

بنابراین، انتخاب تعداد مناسب لایه‌های LSTM بستگی به مجموعه داده‌ها، منابع موجود و مسئله مورد نظر دارد چرا که در شرایط خاص می تواند نتیجه های متفاوتی به ما بدهد.

سوالات تئوری بخش عملی :

به نظر شما افزایش یا کاهش متغیر تعریف شده در مرحله تهیه داده مورد نیاز برای آموزش مدل یعنی تعداد داده های گذشته برای پیش بینی داده های مشخص چه مزایا یا معایبی دارد؟ شرح دهید.

افزایش یا کاهش متغیر n_past که تعداد داده‌های گذشته برای پیش‌بینی داده‌های آینده را مشخص می‌کند، می‌تواند تأثیرات متفاوتی بر روی عملکرد مدل داشته باشد:

مزایا و معایب افزایش n_past:

- **مزیت :** با افزایش n_past، مدل بیشتر از اطلاعات تاریخی استفاده می‌کند که می‌تواند به دقت پیش‌بینی را افزایش دهد. این می‌تواند به خصوص در مواردی که الگوهای زمانی طولانی مدت وجود دارد، مفید باشد.
- **عیب :** افزایش n_past می‌تواند منجر به افزایش زمان آموزش و پیچیدگی محاسباتی شود. همچنین، اگر n_past بیش از حد زیاد شود، ممکن است مدل در نویزهای غیرضروری داده‌ها گرفتار شود که منجر به بیش‌برازش می‌شود.

مزایا و معایب کاهش n_past:

- **مزیت :** کاهش n_past می‌تواند زمان آموزش را کاهش دهد و مدل را ساده‌تر کند. این می‌تواند در مواردی که الگوهای زمانی کوتاه مدت اهمیت دارند، مفید باشد.
- **عیب :** با کاهش n_past، مدل کمتر از اطلاعات تاریخی استفاده می‌کند که ممکن است منجر به کاهش دقت پیش‌بینی شود. این می‌تواند به خصوص در مواردی که الگوهای زمانی طولانی مدت وجود دارد، مشکل‌ساز باشد.

بنابراین، انتخاب بهینه برای n_past بستگی به مشخصات خاص مجموعه داده و مسئله مورد نظر دارد. این مقدار می‌تواند با استفاده از تکنیک‌هایی مانند cross-validation تعیین شود.