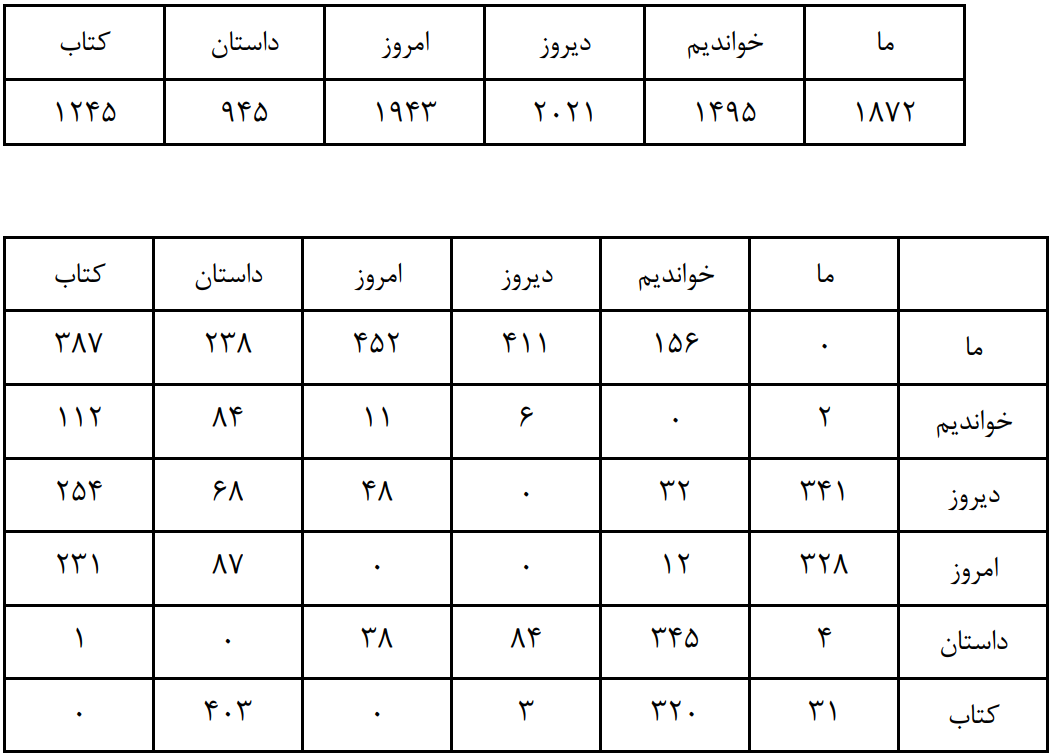
به نام خدا

تكليف سري چهارم NLP بهاره كاوسي نژاد – 99431217

سوالات تئوری

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



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

جمله تست ۱ : ... ما امروز کتاب خواندیم ...

1. ابتدا احتمال رخداد هر کلمه را با توجه به unigram محاسبه می کنیم:

* P(ما) = Count(ما) / Total Count of Unigrams = 1872 / (1872 + 1495 + 2021 + 1943 + 945 + 1245) = 1872 / 9521 = 0.196
* P(امروز) = Count(امروز) / Total Count of Unigrams = 1943 / 9521 = 0.204
* P(کتاب) = Count(کتاب) / Total Count of Unigrams = 1245 / 9521 = 0.130
* P(خواندیم) = Count(خواندیم) / Total Count of Unigrams = 1495 / 9521 = 0.157

1. احتمال رخداد هر bigram را با توجه به جدول bigramها محاسبه می کنیم:

* P(امروز|ما) = Count(ما امروز) / Count(ما) = 452 / 1872 = 0.241
* P(کتاب|امروز) = Count(امروز کتاب) / Count(امروز) = 231 / 1943 = 0.118
* P(خواندیم|کتاب) = Count(کتاب خواندیم) / Count(کتاب) = 320 / 1245 = 0.257

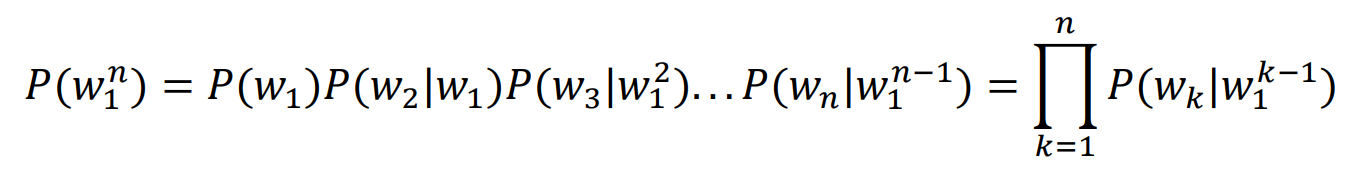
1. احتمال نهایی را با ضرب احتمالات کلمات یا bigramها بدست می آوریم:

* P(ما امروز کتاب خواندیم) = P(ما) \* P(امروز|ما) \* P(کتاب|امروز) \* P(خواندیم|کتاب) = 0.196 \* 0.241 \* 0.118 \* 0.257 = 0.001

جمله تست ۲ : ... ما دیروز داستان خواندیم ...

* ابتدا احتمال رخداد هر کلمه را با توجه به unigram محاسبه می کنیم:
* P(ما) = Count(ما) / Total Count of Unigrams = 1872 / (1872 + 1495 + 2021 + 1943 + 945 + 1245) = 1872 / 9521 = 0.196
* P(دیروز) = Count(دیروز) / Total Count of Unigrams = 2021 / 9521 = 0.212
* P(داستان) = Count(داستان) / Total Count of Unigrams = 945 / 9521 = 0.099
* P(خواندیم) = Count(خواندیم) / Total Count of Unigrams = 1495 / 9521 = 0.157
* احتمال رخداد هر bigram را با توجه به جدول bigramها محاسبه می کنیم:
* P(دیروز|ما) = Count(ما دیروز) / Count(ما) = 411 / 1872 = 0.219
* P(داستان|دیروز) = Count(دیروز داستان) / Count(دیروز) = 68 / 2021 = 0.033
* P(خواندیم|داستان) = Count(داستان خواندیم) / Count(داستان) = 345 / 945 = 0.365
* احتمال نهایی را با ضرب احتمالات کلمات یا bigramها بدست می آوریم:
* P(ما دیروز داستان خواندیم) = P(ما) \* P(دیروز|ما) \* P(داستان|دیروز) \* P(خواندیم|داستان) = 0.196 \* 0.219 \* 0.033 \* 0.365 = 0.0005

1. رابطه زیر را اثبات کنید.

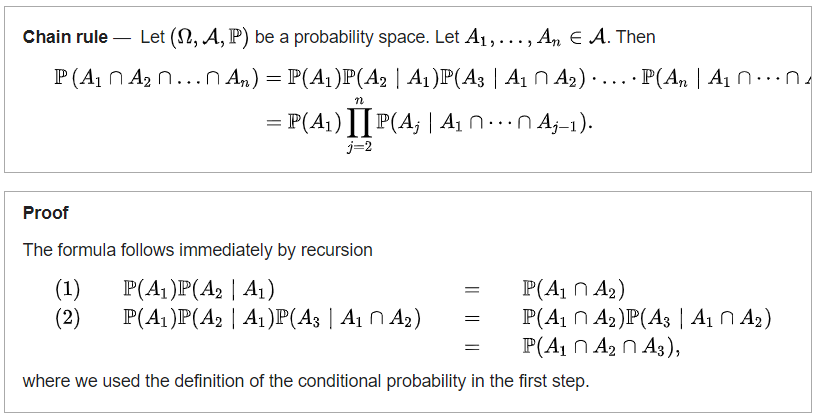


دنباله n کلمه را یا به صورت w1…wn و یا به صورت w1n نمایش می دهیم. بنابراین عبارت w1n-1 به معنای رشته w1, w2, …, wn-1 است. برای joint probability هر کلمه در یک دنباله که به این صورت

P(X1 = w1, X2 = w2, X3 = w3, …, Xn=wn)

نوشته می شود از P(w1,w2,...,wn) استفاده می کنیم.

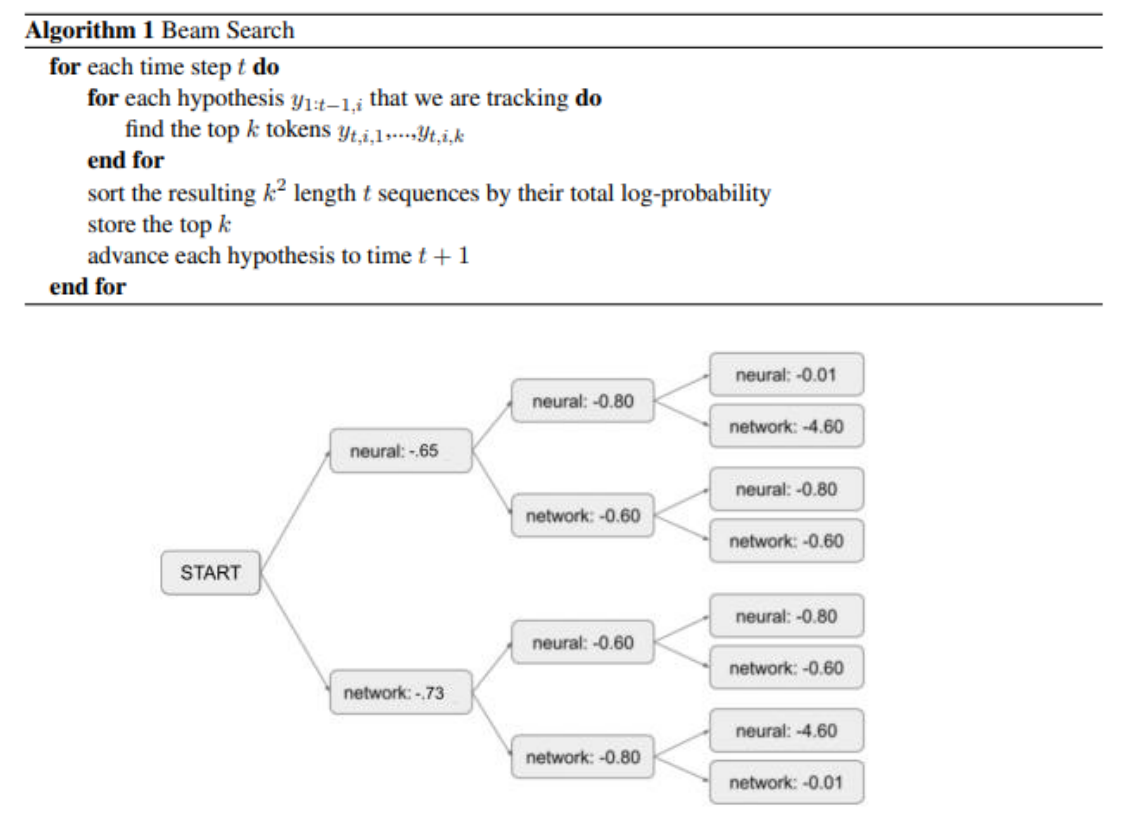
برای محاسبه احتمال کل دنباله مانند P(w1,w2,...,wn) می توانیم از chain rule of probability استفاده کنیم.



با اعمال chain rule به کلمات خواهیم داشت:

Chain rule ارتباط بین محاسبه joint probability یک دنباله را نشان می دهد و محاسبه احتمال شرطی یک کلمه با کلمات قبلی. معادله بالا نشان می دهد که ما می توانیم joint probability یک دنباله کامل کلمات را با ضرب تعدادی احتمال شرطی در یکدیگر تخمین بزنیم.

1. روش search beam را می توان به صورت شبه کد زیر نوشت:



Beam search را برای کدگشایی دنباله ای به طول 3 با k = 2 اجرا میکنیم. پیشبینی های یک کدگشا را در شکل بالا در نظر بگیرید، که در آن هر گره در درخت نشان دهنده پیشبینی احتمال ثبت توکن بعدی مشروط به توکن های قبلی در یک مرحله از کدگشا است (اعداد نشان داده شده، لگاریتم پیشبینی احتمال (log probability prediction) کدگشا از توکن فعلی با توجه به توکنهای قبلی است).

Vocabulary از دو کلمه تشکیل شده است:

“network”, “neural”

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

تنها دو گزینه وجود دارد، پس beam search هر دو را نگه می دارد:

* “neural” (log prob = -.65)
* “network” (log prob = -.73)

1. در مرحله زمانی 2، beam search کدام دنباله را ذخیره می کند؟

تمام دنباله های دو کلمه ای را در نظر میگیریم، اما فقط می توانیم دوتای اول را نگه داریم:

* “neural network” (log prob = -.65 - .6 = -1.25)
* “network neural” (log prob = -.73 - .6 = -1.33)

1. در مرحله زمانی 3، beam search کدام دنباله را ذخیره می کند؟

دنباله های سه کلمه ای را در نظر میگیریم که با “neural network” و “network neural” آغاز می شوند و دوتای اولی به شکل زیر هستند:

* “neural network network” (log prob = -.65 - .6 - .6 = -1.85)
* “network neural network” (log prob = -.73 - .6 - .6= -1.93)

1. آیا search beam دنباله ی کلی با بیشترین احتمال (overall most-likely sequence) در این مثال را برمی گرداند؟ توضیح دهید.

خیر، دنباله کلی با بیشترین احتمال برابر با “neural neural neural” با احتمال زیر است:

log prob = -.65 - .8 - .01= -1.46

این دنباله ها بازگردانده نمی شوند زیرا در مرحله 2 حذف می شوند و در نظر گرفته نمی شوند (زیرا “neural neural” در دنباله ای با بیشترین احتمال و k=2 در دنباله های دو کلمه ای نیست).

1. پیچیدگی زمان اجرا تولید یک دنباله با طول T با اندازه پرتو k با RNN چقدر است؟ برحسب T و k و M پاسخ دهید (M اندازه vocabulary است).

* Step RNN forward one step for one hypothesis = O(M)

زیرا برای هر کلمه در لغات یک logit حساب می کنیم و عملیات های RNN دیگر بر M، T و K وابستگی ندارند.

* Do the above, and select the top k tokens for one hypothesis.

این کار را با sort کردن logitها انجام می دهیم؛ در نتیجه داریم: O(M logM)

راه هایی با پیچیدگی کمتر برای انتخاب K مورد اول وجود دارد (به عنوان مثال استفاده از min heap). از این روش استفاده می کنیم زیرا پیاده سازی آن آسان تر است.

در نتیجه با در نظر گرفتن مرحله قبلی تا کنون به این پیچیدگی رسیده ایم:

O(M log M + M) = O(M log M).

* مراحل بالا را K بار برای current hypotheses انجام می دهیم و داریم: O(KM log M)
* مراحل بالا + انتخاب K مورد اول از K2 تا hypothesesی که اکنون ذخیره شده اند:

این کار را با sort کردن آرایه ای از K2 مورد انجام می دهیم؛ یعنی: O(K2 log(K2)). از آنجایی که log(K2) = 2log(K) می توان پیچیدگی را به این صورت نوشت: O(K2log(K))

(راه هایی با پیچیدگی کمتر نیز برای این مرحله وجود دارند)

با در نظر گرفتن مراحل قبل تر داریم: O(K M log M + K2 log (K))

از آنجایی که M بزرگتر مساوی K است می توان نوشت: O(K M log M)

این کار را برای T تا timestep انجام می دهیم: O(T K M log M)

**منابع:**

* <https://en.wikipedia.org/wiki/Chain_rule_(probability)>
* <https://inst.eecs.berkeley.edu/~cs182/sp23/assets/section/dis07/sol07.pdf>