سبحان رنجبر

من در کل موارد تنها از مقالات که در زمینه word2vec می باشد استفاده کردم.

سوال 1)

1.1

fox, dog => \b(dog|fox)\b

\b برای این که کلمات مستقل پیدا کند و | معنی همان <<یا>> را می دهد.

1.2

\b[d|f]\w*[g|x]\b

d|f به معنی d یا f

\w* هر تعداد کلمه می تواند این وسط باشد.

1.3

\b\w*[aeiou]{2,}\w*\b

\w*\b, \b\w*

فقط اگر کلمه باشد بررسی می کند.

[aeiou]{2,}

حداقل دو حرف صدادار در یک کلمه می گردد

1.4

\b(\w)\w*\1\b

(\w):

اولین حرف از کلمه را در گروهی ذخیره می‌کند

\w*:

شامل حروف داخلی کلمه می‌شود

\1:

.تطبیق مجدد با همان حرف اول

1.5

\b[lL]\w*\b([?!.]=?)

\b[lL]:

کلماتی که با "l" یا "L" شروع می‌شوند.

\w*:

ادامه‌ی کلمه را می‌گیرد

\b(?=[.!?]):

تضمین می‌کند که این کلمه در انتهای جمله باشد

سوال 2)

در دنیای واقعی ما چون اشکال مختلف فعل و اسم و قید و ... داریم برای محاسبه هرکدام و ساخت بردار های ان نیاز به فضای زیادی داریم و از لحاظ معنایی یک فعل و شکل های مختلف ان فرق ندارد پس می اییم از دو روش Stemming و Lemmatization استفاده می کنیم.

Lemmatization:

فرایند دقیق تر است که کلمات را به ریشه لغوی (lemma) بر می گردانند.با استفاده از روش فرهنگ لغت و قواعد دستور زبان استفاده می کند تا شکل اصلی کلمه را پیدا کنند.

Stemming:

پسوند ها و پیشوند ها یک کلمه را حذف می کنیم تا به stemming برسیم. از قواعد ساده استفاده می کنند و تبدیل به ریشه خام می کند که شاید کلممه معنی دار نباشد.

Ex)

Lemmatization => خوردم -> خوردن

Stemming => خوردم -< خورد

سوال 3)

king → [0.1,−0.9,0.2]

queen → [0.7,0.5,0.3]

apple → [0.8,0.6,−0.4]

apple → [0.8,0.6,−0.4] این درست می باشد چون کلمه King , Queen هردو مربوط به سلطننت می باشند و انسان می باشند و به هم نزدیک تر می باشند و apple از این دو دورتر می باشد.

سوال 4)

روش Bigram یعنی احتمال آمدن کلمات پشت سر هم. (به صورت دوتایی)

حتما باید یکی تعداد کلمات به صورت تنهایی بدست بیاوریم. ← V

احتمال با bigram Add-one smathing

$$P(w_i | w_{i-1}) = \frac{C(w_{i-1} w_i) + 1}{C(w_{i-1}) + V}$$

C → تعداد حضور کلمه

V → شمارش کلمات به صورت منحصر به فرد

$$P(\langle s \rangle \text{the lazy fax jumps} \langle /s \rangle) =$$

$$= P(\text{the} | \langle s \rangle) \times P(\text{lazy} | \text{the}) \times P(\text{fox} | \text{lazy}) \times$$

$$\times P(\text{fox} | \text{lazy}) \times P(\text{jumps} | \text{fox}) \times P(\langle /s \rangle | \text{jumps})$$

★ در این جا کلی مورد مقایسه می کنیم و بقیه به همین گونه می باشد.

$$P(\text{the} | \langle s \rangle) = \frac{C(\langle s \rangle \text{the}) + 1}{C(\langle s \rangle) + V} = \frac{1+3}{10+5} = \frac{4}{15}$$

V ↙ ، C(<s>) ↘

$$\text{answer} = \frac{1}{13} \times \frac{4}{13} \times \frac{1}{13} \times \frac{1}{4} \times \frac{4}{15} = \frac{16}{13^3 \times 5}$$

سوال 5)

در این سوال قرار است از صفر word2vec را پیاده سازی کنیم در این جا چند مورد را باید بلد باشیم مثل این که چگونه از صفر مدل یادگیری عمیق را بنویسیم و قوانین پیاده سازی ان مثل forwardpass چگونه به جلو بریم در مدل و یا backprop که چگونه مدل را اپدیت کنیم و کلیت کار ما به این شکل است داده ها را بگیرم چون کلمات ما از نوع رشته ای است باید تبدیل به بردارد عددی کنیم که راحت ترین ان تبدیل به وان هات می باشد و بعد از تبدیل به بردار عدد ان را به مدل یادگیری عمیق بدهیم و روی ان اموزش ببیند (train)

در اخر هم می توانیم از activtion function مختلف استفاده کنیم که به صورت پیشفرض در word2vec از softmax استفاده می شود که داخل پرانتز در مقالات بعدی سعی شد از این تابع استفاده نشود چون درمخرج کسر باید همه موارد هر دفعه بررسی می کرد که هزینه محاسبات داشت.

و در اخر برای پیدا کردن شباهت کلمات می توانیم از cosinesimilarity و یا دوستاش استفاده کنیم و ان هایی که بردار نزدیک تری دارند ان را انتخاب کنیم.

سوال 6)

در این سوال با کتابخونه gensim قرار بود هر دو مدل wod2vec را برررسی کنیم و با هم مقایسه کنیم , skip-gram Cbow پالشی که من تو این سوال داشتم پیدا کردن درست استفاده کردن از لایبرری بود داک خیلی قوی نداشت و دیگری من سعی کردم کلمه به کلمه بخونم که رم 16 توانایش نداشت و از تی ای هم پرسیدم و فهمیدم باید جمله به جمله بخونم.

براساس مقاله خود wod2vec و نتایج که رسیدیم skip-gram راه حل بهتری می باشد.