۱) الف) تفاوت ۱

در روش NNLM بردارهای کلمات بصورت nonlinear با یکدیگر جمع می‌شوند یعنی بین اصطلاحاتی که با تغییر ترتیب کلمات، تغییر معنای خیلی زیادی دارند تفاوت در نظر گرفته می‌شود درحالی که در روش Continous bag of words صرفا جمع انجام می‌دهیم و اصلا به مسائل ترتیب توجه نداریم.

تفاوت ۲

در روش continous bag of words با پنجره‌ای از کلمات را استفاده می‌کنیم و سعی می‌کنیم هم به چپ و هم به راست توجه کنیم درحالی که در NNLM به کلمات بعدی توجهی نداریم و صرفا به کلمات قبل توجه داریم.

مثلا من در سینما یکی از انواع غذای تند و تیز را ترجیح می‌دهم.

یا من در سینما یکی از انواع غذای شیرین را ترجیح نمی‌دهم.

در اینجا کلمه غذا هیچ تفاوتی در هر دو context ندارد زیرا صرفا به قبل توجه داریم.

۱) ب)

merge of word vectors → $N \times D$

Calculating h → $N \times D \times h$

Calculating $\hat{y}$ from h → $H \times V$

$\Rightarrow O(NDH + HV) \xrightarrow{V \gg ND} O(HV)$

۱) ج)

می‌توانیم به جای محاسبه softmax از Negative sampling استفاده کنیم یا می‌شود از hierarchical softmax استفاده کرد.

۱) د)

$$z_1 = xW + b \qquad \delta_1 = \hat{y} - y$$
$$z_2 = hU + d \Rightarrow \delta_2 = \delta \frac{\partial z_2}{\partial h} = U^T \delta_1$$
$$\delta_3 = \delta_2 \frac{\partial h}{\partial z_1} = \delta_2 (1 - \tanh^2(z_1))$$
$$\Rightarrow \frac{\partial CE}{\partial x} = \delta_3 \frac{\partial z_1}{\partial x} = W^T \delta_3$$

۲) الف)

معایب روش word2vec: سرعت کند و پیچیدگی محاسباتی بیشتر در حالی که روش CCBM از لحاظ هزینه محاسباتی بزرگتر بهینه‌تر عمل می‌کند

مزایای word2vec: می‌تواند الگوهای پیچیده‌تر را استخراج کند و به سادگی نیز scalable هست.

ب) خیر [illegible] این مسئله بدیهی نیز به سادگی مشخص است لزوما وقتی ضرب دو بردار یکی شود به معنای یکسان بودن بردارها نیست می‌توان به سادگی بردارها را یک عدد m و $\frac{1}{m}$ ضرب کرد یا می‌توان آن را به اندازه α درجه rotate کرد

ج) چون هزینه محاسباتی بالا می‌رود (اگر سایز V یا همان vocabulary خیلی زیاد باشد) ⇐ راهکار

(۱) Negative Sampling

(۲) Hierarchical Softmax

۲) د) Glove >> word2vec

از لحاظ حافظه Glove نیز حتی مصرف بیشتری دارد علت آن نیز این است که Glove روی CCBM (concurrence count-based methods) ها آموزش داده می‌شود.

۲) ۵) اولاً هیچ کدام توانایی تشخیص کلمات متضاد را ندارند البته علت آن این است که کلمات متضاد در فضای vector بسیار به یکدیگر نزدیک هستند چون اغلب در contextهای مشابه یافت می‌شوند.

ثانیاً در یادگیری کلمات خارج زبان training بسیار ضعیف هستند و فقط روی کلماتی که در زبان train بوده خوب عمل می‌کنند.

۲) و) محدب نیست زیرا اولاً اگر به صورت یکسان بردارها را (بردارهای تعبیه) permute کنیم تابع هزینه یکسان خواهد بود

مثلاً فرض کنید از embedding vectorها میانگین بگیریم این باعث می‌شود که تمام مقادیر تمام ابعاد یکسان باشد و هزینه افزایش می‌یابد ⟸ محدب نیست (زیرا اگر محدب می‌بود کاهش داشتیم)

ثانیاً فرض کنید R و $\tilde{R}$ را میانگین بگیریم بازهم هزینه افزایش می‌یابد که معلوم می‌شود محدب نیست به صورت کلی اگر R و $\tilde{R}$ مستقیماً تغییر داده شوند هزینه کاهش نمی‌یابد.

۳) الف) در این رویکرد برای بدست آمدن embedding از بردار کلمه ورودی و position آن تجمیع می‌کنیم و در نتیجه position کلمه حفظ می‌شود یعنی یک کلمه در position های مختلف embedding های متفاوتی خواهد داشت.

ب) در انتقال دهنده ها (Transformer) در واقع embedding قبل از وارد شدن به encoder با ماتریس position ترکیب شده که باعث می‌شود مکان حفظ شود.

ج) در مدل های Encoder Decoder تمام معنا و مفهوم جمله input (یا همان بردار hidden) باید در یک بردار ذخیره شود بنابراین هرچه اندازه جمله ورودی بزرگتر شود نگهداری اطلاعات سخت‌تر می‌شود و نهایتاً در یک threshold اطلاعات دور ریخته می‌شود که به آن فراموشی گویند ولی در مکانیزم توجه قسمت های مهم جمله ورودی به شبکه شناسانده می‌شود و توجه بیشتری به آن می‌شود (البته بعضاً به جای vector نیز از matrix استفاده می‌کنند)