HW3 Word Embeddings

Jing Qian (jq2282)

1. Parameter search

(1) Table of the parameter search results

Algorithm	Win.	Dim.	N.s.	WordSim	BATS 1 (encyclopedic-semantics	BATS 2 (antonyms - binary)	BATS 3 (total)	MSR
word2vec	2	100	1	0.055	0.033	0.023	0.013	0.661
word2vec	2	100	5	0.197	0.030	0.070	0.020	0.678
word2vec	2	100	15	0.222	0.062	0.093	0.029	0.674
word2vec	2	300	1	0.098	0.037	0.093	0.019	0.665
word2vec	2	300	5	0.198	0.022	0.116	0.019	0.669
word2vec	2	300	15	0.223	0.049	0.186	0.040	0.672
word2vec	2	1000	1	0.025	0.049	0.047	0.023	0.667
word2vec	2	1000	5	0.205	0.057	0.163	0.029	0.673
word2vec	2	1000	15	0.225	0.061	0.163	0.031	0.678
word2vec	5	100	1	0.149	0.038	0.047	0.018	0.664
word2vec	5	100	5	0.233	0.050	0.047	0.027	0.678
word2vec	5	100	15	0.264	0.034	0.163	0.026	0.679
word2vec	5	300	1	0.122	0.044	0.0	0.015	0.666
word2vec	5	300	5	0.227	0.041	0.070	0.022	0.677
word2vec	5	300	15	0.281	0.054	0.093	0.030	0.675
word2vec	5	1000	1	0.124	0.042	0.047	0.016	0.659
word2vec	5	1000	5	0.228	0.039	0.140	0.028	0.675
word2vec	5	1000	15	0.272	0.032	0.070	0.026	0.677
word2vec	10	100	1	0.168	0.061	0.047	0.021	0.674
word2vec	10	100	5	0.287	0.050	0.023	0.028	0.674
word2vec	10	100	15	0.303	0.032	0.163	0.028	0.673
word2vec	10	300	1	0.184	0.017	0.070	0.012	0.666
word2vec	10	300	5	0.279	0.024	0.163	0.026	0.675
word2vec	10	300	15	0.310	0.043	0.070	0.024	0.672
word2vec	10	1000	1	0.173	0.035	0.116	0.015	0.667
word2vec	10	1000	5	0.267	0.048	0.093	0.023	0.676
word2vec	10	1000	15	0.291	0.029	0.116	0.018	0.668
SVD	2	100	-	0.162	0.039	0.044	0.021	0.648
SVD	2	300	-	0.252	0.007	0.111	0.017	0.639
SVD	2	1000	-	0.311	0.0	0.089	0.008	0.643
SVD	5	100	-	0.266	0.031	0.156	0.026	0.648
SVD	5	300	-	0.346	0.018	0.067	0.010	0.643
SVD	5	1000	-	0.302	0.0	0.022	0.005	0.650
SVD	10	100	-	0.366	0.021	0.2	0.024	0.651
SVD	10	300	-	0.402	0.010	0.0	0.008	0.651
SVDs	10	1000	-	0.339	0.0	0.0	0.001	0.656

(2) Written analysis of the results

i) Does larger dimensionality always equal to better performance? In which categories and for which models? Why do you think this is?

No, larger dimensionality doesn't always equal to better performance.

For SVD model, when window size = 2, increasing the dimensionality increases the WordSim wheras window size = 5 or 10, increasing the dimentionality from 100 to 300 increases the WordSim and that from 300 to 1000 actually decreases the WordSim. For three BATS, increasing the dimensionality almost decreases all BATS values. MSR increases when increasing dimentionality from 100 to 300 and decreases when increasing dimentionality from 300 to 1000. I think maybe due to the small size of our training corpus, increasing the dimentionality (which is the k in SVD truncation) from 100 to 300 does introduce more information in the word embeddings. However, when increasing from 300 to 1000, more noises are conserved in the word embeddings, which leads to the decreasement in all metrics.

For word2vec SGNS model, following combinations of hyperparameters have certain categories have better performance with increasing dimensionality:

window size = 2, NS = 1, BATS1, BATS3, MSR.

window size = 2, NS = 5, wordSim, BATS2

window size = 2, NS = 15, wordSim

window size = 5, NS = 5, BATS2

window size = 10, NS = 1, BATS2

window size = 10, NS = 5, MSR

Other hyperparameter combinations or categories don't have the increasing trend with larger dimensionality. We could see that with smaller window size and smaller number of negative samples, more categories show the increasing trend. I think it is due to the similar analysis for the SVD case, which is the limited training corpus size. With large window size and large negative sample sizes, if we also increase the dimensionality, we would overfit to the noise of our training corpus. Also, different categories show different performance changing behavior with the change of hyperparameters.

ii) Does better performance on one task mean better performance on the others? Provide a hypothesis as to why or why not?

No, better performance on one task does not necessarily mean better performance on the others. Because the three tasks here evaluate the word embeddings in three different aspects: WordSim353 evaluates the word similarity with cosine similarity and Spearman's ρ, the BATS did analogy prediction on various aspects and MSR performs paraphrase detection using logistic regression over cosine similarity scores.Even among different categories of BATS, the performance vary across the categories. For example, one word embeddings may have high word similarity from the WordSim353 but performs low on the MSR paraphrase task, like when window size=2, dim. = 100, changing N.s. from 5 to 15, the WordSim increases from 0.197 to 0.222 while the MSR decreases from 0.678 to 0.674.

iii) Was performance roughly similar across all analogy categories? If different, how did it vay? Why do you think you observed this variation? Perform a brief error analysis and compare erros across the BATS categories you selected for your table.

No, performance is not roughly similar across all analogy categories. It is affected by the hyperparameters. For SVD model, when increasing the dimensionality, except BATS 2, other BATS decreases. For word2vec model, for some hyperparameters (win = 2, dim = 100, ns in (1,5,15)), the performance are the same. But for others, are not. I think it also depends on the categories that I choose. For example, when increasing the window size, some categories may have a larger performance change, like the BATS 2(antonyms - binary), however others may not, then the BATS 3(total) got even.

2. Fun with objective functions.

1) (Preliminaries)

i)

$$\sigma(-x) = \frac{1}{1+e^{x}} = \frac{e^{-x}}{1+e^{-x}} = 1 - \frac{1}{1+e^{-x}} = 1 - \sigma(x).$$

ii) Using Chain rule, we could do the derivatives:

$$\frac{d}{dx}\sigma(x) = -\frac{1}{(1+e^{-x})^2}\frac{d}{dx}e^{-x} = \frac{e^{-x}}{(1+e^{-x})^2} = \sigma(x)(1-\sigma(x)).$$

iii) Using Chain rule, we could do the derivatives:

$$\frac{d}{dx}\log(\sigma(x)) = \frac{1}{\sigma(x)}\frac{d}{dx}\sigma(x) = \frac{1}{\sigma(x)}\sigma(x)(1-\sigma(x)) = 1-\sigma(x).$$

2) (A simpified global objective)

i) The inner expectation in the SGNS loss function as a sum is:

$$\mathbb{E}_{c'\sim Pn(c)}[\log\sigma(-\vec{w}\cdot\vec{c}')] = \sum_{c'\in V_c} P_n(c')\cdot\log\sigma(-\vec{w}\cdot\vec{c}') = \sum_{c'\in V_c}\frac{N_{c'}}{N}\cdot\log\sigma(-\vec{w}\cdot\vec{c}').$$

ii) Since $\sum_{c\in V_c} N_{w,c} = N_w$, we have:

$$\begin{aligned}
L &= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}(\log\sigma(\vec{w}\cdot\vec{c}) + k\cdot\mathbb{E}_{c'\sim Pn(c)}[\log\sigma(-\vec{w}\cdot\vec{c}')]) \\
&= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_w}(\sum_{c\in V_c} N_{w,c})k\cdot\mathbb{E}_{c'\sim Pn(c)}[\log\sigma(-\vec{w}\cdot\vec{c}')] \\
&= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_w} N_w k\cdot\mathbb{E}_{c'\sim Pn(c)}[\log\sigma(-\vec{w}\cdot\vec{c}')] \\
&= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_w} N_w k\cdot\sum_{c'\in V_c}\frac{N_{c'}}{N}\cdot\log\sigma(-\vec{w}\cdot\vec{c}') \\
&= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_w} N_w k\cdot\sum_{c\in V_c}\frac{N_c}{N}\cdot\log\sigma(-\vec{w}\cdot\vec{c}) \\
&= \sum_{w\in V_w}\sum_{c\in V_c} N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_w}\sum_{c\in V_c} k\cdot N_w\frac{N_c}{N}\cdot\log\sigma(-\vec{w}\cdot\vec{c}) \\
&= \sum_{w\in V_w}\sum_{c\in V_c}[N_{w,c}\log\sigma(\vec{w}\cdot\vec{c}) + k\cdot N_w\frac{N_c}{N}\cdot\log\sigma(-\vec{w}\cdot\vec{c})].
\end{aligned}$$

3) (Optimizing at the local level)

i) If we take $x = \vec{w}\cdot\vec{c}$, we have:

$$l = N_{w,c}\log\sigma(x) + k\cdot N_w\frac{N_c}{N}\cdot\log\sigma(-x).$$

And using the derivatives from part 1), the derivative of l with respect to x is:

$$\begin{aligned}
\frac{d}{dx}l &= N_{w,c}\frac{d}{dx}\log\sigma(x) + k\cdot N_w\frac{N_c}{N}\cdot\frac{d}{dx}\log\sigma(-x) \\
&= N_{w,c}(1-\sigma(x)) + k\cdot N_w\frac{N_c}{N}\cdot\frac{d}{dx}\log(1-\sigma(x)) \\
&= N_{w,c}(1-\sigma(x)) - k\cdot N_w\frac{N_c}{N}\cdot\sigma(x) \\
&= N_{w,c} - (k\cdot N_w\frac{N_c}{N} + N_{w,c})\cdot\sigma(x)
\end{aligned}$$

ii) Setting $\frac{d}{dx}l = 0$, we have:

$$
\begin{aligned}
&N_{w,c} - (k \cdot N_w \frac{N_c}{N} + N_{w,c}) \cdot \sigma(x) = 0, \\
&\sigma(x) = \frac{N_{w,c}}{k \cdot N_w \frac{N_c}{N} + N_{w,c}} = \frac{1}{1 + k \cdot \frac{N_w N_c}{N N_{w,c}}}, \\
&\frac{1}{1+e^{-x}} = \frac{1}{1 + k \cdot \frac{N_w N_c}{N N_{w,c}}}, \\
&x = -\log(k \cdot \frac{N_w N_c}{N N_{w,c}}) = -\log(\frac{N_w N_c}{N N_{w,c}}) - \log k = \log(\frac{N_{w,c} N}{N_w N_c}) - \log k.
\end{aligned}
$$

iii) Since $x = \vec{w} \cdot \vec{c}$, the optimal $\vec{w} \cdot \vec{c}$ corresponds to the optimal x in part 3) ii). Also, according to the definition of $PMI(w, c)$, we have:

$$
\vec{w} \cdot \vec{c} = x = \log(\frac{N_{w,c} N}{N_w N_c}) - \log k = PMI(w, c) - \log k.
$$