정보검색 과제 2: word2vec

Abstract

본 과제의 목표는 `word2vec`을 실제로 구현함으로서 자연어 처리에 대한 이해를 높이는 것입니다. 강의를 듣는 것과 실제로 구현하는 것에는 큰 차이가 있습니다. 본 과제는 스탠포드 자연어 처리 강의 (cs224n)에서 차용하였음을 미리 밝힙니다 [1]. 과제가 쉽다고는 말하지 못하겠습니다만, 할 수 없는 것이라고 생각하지는 않습니다. 본 과제에서 가장 중요한 재료는 시간과 끈기일 것입니다. 본 과제를 수행하면서 보람과 성취감을 얻길 바랍니다. 문의사항은 언제든 조교에게 연락하기 바랍니다.

1 서술형: word2vec 이해

`word2vec` 알고리즘을 복습해보자. `word2vec`의 핵심은 *'a word is known by the company it keeps'* 이다. 구체적으로, 'center' word c 와 c 주변의 contextual window 를 가정해보자. 이 contextual window 에 있는 단어들을 'outside words'라고 하자. 예를 들면 Figure 1 에서, center word c 는 'banking' 이다. context window 크기가 2 이므로, outside words 들은 'turning', 'into', 'crices', 'as' 가 된다.

Skip-gram `word2vec` 의 목표는 확률분포 $P(O|C)$ 를 정확히 학습하는 것이다. 즉, 주어진 특정 단어 o 와 특정 단어 c 에 대하여 $P(O = o|C = c)$ 를 얻고자 한다. 이 때 $P(O = o|C = c)$ 는 단어 c 에 대하여 단어 o 가 'outside' word 일 확률을 의미한다. 다시 말해, o 가 c 의 contextual window 안에 있을 확률을 말한다.

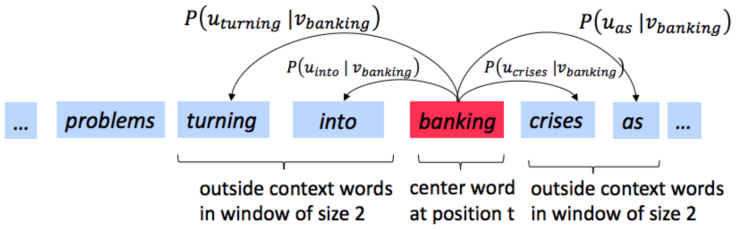

Figure 1: 윈도우 크기가 2일때 word2vec skip-gram 예측 모델

`word2vec` 에서, 조건확률분포는 벡터의 내적과 softmax 함수의 적용으로 주어진다.

$$P(O = o|C = c) = \frac{\exp(\boldsymbol{u}_o^\top \boldsymbol{v}_c)}{\sum_{w \in \text{Vocab}} \exp(\boldsymbol{u}_w^\top \boldsymbol{v}_c)} \tag{1}$$

여기서, $\boldsymbol{u}_o$ 는 outside word o 를 나타내는 'outside' 벡터를, $\boldsymbol{v}_c$ 는 center word c 를 나타내는 'center' 벡터를 의미한다. 이 파라메터를 얻기위해, $\boldsymbol{U}$ 와 $\boldsymbol{V}$ 의 두

[1] `http://web.stanford.edu/class/cs224n/`

행렬을 이용한다. $\boldsymbol{U}$ 의 열은 'outside' 벡터 $\boldsymbol{u}_w$ 이다. $\boldsymbol{V}$ 의 열은 'center' 벡터 $\boldsymbol{v}_w$ 이다. $\boldsymbol{U}$ 와 $\boldsymbol{V}$ 는 모든 $w \in$ Vocabulary 의 벡터를 포함한다.

c 와 o 의 단어쌍에 대하여, loss 는 다음과 같이 주어짐을 기억하자.

$$\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, o, \boldsymbol{U}) = -\log P(O = o|C = c). \tag{2}$$

이 loss 는 true distribution $\boldsymbol{y}$ 와 $\hat{\boldsymbol{y}}$ 간의 cross-entropy로 볼 수도 있다. 여기서, $\boldsymbol{y}$ 와 $\hat{\boldsymbol{y}}$ 는 vocabulary에 있는 단어의 수에 해당하는 길이를 갖는 벡터이다. 나아가서, 이 벡터들의 k^{th} 번째 entry는 vocabulary 의 k^{th} 번째 단어가 주어진 c 에 대하여 'outside word' 가 될 조건부 확률을 의미한다. 따라서 true empirical distribution $\boldsymbol{y}$ 는 true outside word o 에 대해서만 1 이고 다른 모든 단어에 대해서는 0인 one-hot 벡터가 된다. Predicted distribution $\hat{\boldsymbol{y}}$ 는 수식 (1) 에 의해 주어진 확률 분포가 된다.

(모든 문제는 첨부한 CS224n Assignment와 동일합니다. *다만, naive-softmax에 해당하는 부분만 작성합니다. negative sampling은 다루지 않습니다.* 모든 해답은 풀이과정이 명확히 기술되어야 합니다.)

(a) naive-softmax loss 가 $\boldsymbol{y}$ 와 $\hat{\boldsymbol{y}}$ 간의 cross-entropy loss 와 같음을 보여라. 즉,

$$-\sum_{w \in \text{Vocab}} y_w \log(\hat{y}_w) = -\log(\hat{y}_o) \tag{3}$$

임을 보여라.

(b) $\boldsymbol{v}_c$ 에 대한 $\boldsymbol{J}$의 편미분을 구하라. 즉

$$\frac{\partial \boldsymbol{J}_{\text{naive-softmax}}}{\partial \boldsymbol{v}_c}$$

를 구하라. 답은 $\boldsymbol{y}$, $\hat{\boldsymbol{y}}$, $\boldsymbol{U}$ 를 이용하여 표현되어야 한다.

(c) $\boldsymbol{u}_w$ 에 대한 $\boldsymbol{J}$의 편미분을 구하라. 즉

$$\frac{\partial \boldsymbol{J}_{\text{naive-softmax}}}{\partial \boldsymbol{u}_w}$$

를 구하라. $w = o$ 인 경우와 $w \neq o$ 인 경우가 있음에 주의하시오. 답은 $\boldsymbol{y}$, $\hat{\boldsymbol{y}}$, $\boldsymbol{v}_c$ 를 이용하여 표현되어야 한다.

(d) x가 scalar 일 때, x 에 대한 시그모이드 함수 $\sigma(x)$의 편미분을 구하라. 시그모이드 함수는 다음과 같이 정의된다.

$$\sigma(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1} \tag{4}$$

답은 $\sigma(x)$ 를 이용하여 작성하시오.

(e) Negative Sampling 은 풀지 않습니다.

$$\boldsymbol{J}_{\text{neg-sample}}(\boldsymbol{v}_c, o, \boldsymbol{U}) = -\log(\sigma(\boldsymbol{u}_o^\top \boldsymbol{v}_c)) - \sum_{k=1}^{K} \log(-\sigma(\boldsymbol{u}_k^\top \boldsymbol{v}_c)) \tag{5}$$

(f) Center word가 $c = w_t$ 이고 context window 가 $[w_{t-m}, \ldots, w_{t-1}, w_t, w_{t+1}, \ldots, w_{t+m}]$ 으로 주어졌다고 가정하자. 이때 m 은 context window size 이다. `word2vec` 을 skip-gram 으로 구현할 때, context window 에 대한 total loss 는 다음과 같음을 상기하자.

$$\boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots, w_{t+m}, \boldsymbol{U}) = \sum_{\substack{-m \le j \le m \\ j \neq 0}} \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) \quad (6)$$

$\boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$는 $\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$ 으로 가정하자 (참고: Negative Sampling 의 loss 도 같은 방법으로 구할 수 있다) . 이 때 $\boldsymbol{J}_{\text{skip-gram}}$의 $\boldsymbol{U}$, $\boldsymbol{v}_c$, $\boldsymbol{v}_w$ 에 대한 편미분을 각각 구하라.

(i) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots, w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{U}$

(ii) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots, w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{v}_c$

(iii) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots, w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{v}_w$ when $w \neq c$

답은 $\partial \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) / \partial \boldsymbol{U}$ 와 $\partial \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) / \partial \boldsymbol{v}_c$ 를 이용하여 기술하라.

2 코드: word2vec 구현

주의 : 코드 템플릿은 과제에 첨부된 것으로 하기 바랍니다. 본 과제에서는 Negative Sampling을 다루지 않으므로, stanford code를 사용하면 not implemented error 가 발생할 수 있습니다.

(a) `word2vec.py` 를 구현해보자. 먼저, `sigmoid` 메소드를 구현하라. 다음으로 `naiveSoftmaxLossAndGraient` 메소드를 구현하라. 마지막으로, `skipgram` 메소드를 구현하라.

(b) `sgd.py`의 `sgd` 메소드를 구현하라.

(c) `python run.py`를 수행하여 `word_vectors.png` 를 플롯하고 결과를 살펴보자.

3 제출방법

클래스룸 제출.