word2vec Models *

Khanh Nguyen

June 11, 2015

There are two *word2vec* models: skip-gram and continuous bag-of-words (CBOW). They both manage to capture interactions between a centered word and its context words. However, they do it differently, and somehow oppositely. While skip-gram models the distribution of context words given the centered word, CBOW is concerned about predicting the centered word given its context.

In this note, I will make no distinction in notation between a word and its corresponding vector. For example, w denotes both the word and its word vector. I treat distinct word vectors as distinct words.

1 General model

Given a predicted word vector $\hat{r}$ and a target word vector w_t. The probability of the target word conditional on the predicted word is calculated by a softmax function:

$$P(w_t|\hat{r}) = \frac{exp(w_t^T \hat{r})}{\sum_{w \in W} exp(w^T \hat{r})}$$

where W is the set of all target word vectors.

Notice that $\hat{r}$ is neither an element of W nor computed from elements of W. As we will see later, elements of W will be called *output vectors* and $\hat{r}$ will be computed from a different set consisting of *input vectors*.

word2vec models' cost functions (for one target word) minimize the negative log-likelihood of the target word vector given its corresponding predicted word:

$$\mathcal{L}(w_t, \hat{r}) = -\log P(w_t|\hat{r}) = \log \left(\sum_{w \in W} exp(w^T \hat{r}) \right) - w_i^T \hat{r}$$

The gradient with respect to w of $\mathcal{L}(w_t, \hat{r})$ is:

$$g_1(w, w_t, W, \hat{r}) = \frac{\partial}{\partial w} \mathcal{L}(w_t, \hat{r}) = \hat{r}(P(w|\hat{r}) - I\{w = w_t\}) \tag{1}$$

where $I\{.\}$ is the indicator function.

*This material is for self-study only.

The gradient with respect to $\hat{r}$ is:

$$g_2(w_t, W, \hat{r}) = \frac{\partial}{\partial \hat{r}} \mathcal{L}(w_t, \hat{r}) = \sum_{w \in W} [P(w|\hat{r})w] - w_t \tag{2}$$

2 Skip-gram

For an index i and a window size c, skip-gram predicts the context words $\{w_j\}$, $(i - c \leq j \leq i + c, j \neq i)$ given the centered word r_i. Hence, in the general model, $w_t = w_j$ and $\hat{r} = r_i$ for this case.

The cost function is derived as follows:

$$\mathcal{L}_{skipgram}(c, i) = \sum_{i-c \leq j \leq i+c, i \neq j} -\log P(w_j|r_i)$$

The gradients of this function are:

$$\frac{\partial}{\partial w} \mathcal{L}_{skipgram}(c, i) = r_i \sum_{i-c \leq j \leq i+c, i \neq j} g_1(w, w_j, W, r_i) \tag{3}$$

$$\frac{\partial}{\partial r_i} \mathcal{L}_{skipgram}(c, i) = \sum_{i-c \leq j \leq i+c, i \neq j} g_2(w_j, W, r_i) \tag{4}$$

3 Continuous bag-of-words

Intuitively, this model reverses the modeling mechanism of skip-gram. CBOW predicts a word given its context. The target word vector is now the output vector of the word at index i, w_i; the predicted word vector is the sum over all context input vectors:

$$\hat{r} = \sum_{i-c \leq j \leq i+c, i \neq j} r_j$$

The CBOW's cost function is as follows:

$$\mathcal{L}_{CBOW}(c, i) = -\log P(w_i|\hat{r})$$

and the gradients are: student

$$\frac{\partial}{\partial w} \mathcal{L}_{CBOW}(c, i) = g_1(w, w_i, W, \hat{r}) \tag{5}$$

$$\frac{\partial}{\partial r_j} \mathcal{L}_{CBOW}(c, i) = g_2(w_i, W, \hat{r}) \tag{6}$$

for $i - c \leq j \leq i + c, i \neq j$. Otherwise,

$$\frac{\partial}{\partial r_j} \mathcal{L}_{CBOW}(c, i) = 0 \tag{7}$$

4 Negative sampling

Read word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method (Goldberg and Levy, 2014) for the motivation of negative sampling.

5 Compare with GloVe

GloVe is a more recent word embedding model developed by CITE Pennington. In the paper of GloVe, the model was compared directly to *word2vec*. There was a debate about how that experiment was set up. The final version of the paper still claimed that *GloVe* outperformed *word2vec* on the task of word analogy (created by CITE Mikolov) if we let it run long enough. I really the discussion on the motivation of the model since very few papers I have read have that kind of meticulous discussion. The authors illustrated how the training objective of *skip-gram* uses information from the occurrence matrix of a corpus, which explains why the model, although only uses local context window, can produce word embeddings that capture global relationships. Hence, the *skip-gram* model can also be viewed as a least square problem where only simple matrix decomposition is required to compute the solution. Thus *GloVe* yields a better performance than *skip-gram*.

Concretely, let X be the occurrence matrix of the corpus where the (i, j) entry is number of times word i and word j co-occur. Denote by $P_{ij} = \frac{X_{ik}}{X_i}$ the probability that word i and word j co-occur. The relationship of word i and word j is determined based on the ratio of the probabilities of their co-occurrences with a context word k:

$$F\left((w_i - w_j)^T w_k^{'}\right) = \frac{P_{ik}}{P_{jk}} \tag{8}$$

Since in the occurrence matrix, the roles of the rows and columns are interchangeable, we enforce symmetric to the above equation:

$$F\left((w_i - w_j)^T w_k^{'}\right) = \frac{F(w_i^T w_k^{'})}{F(w_j^T w_k^{'})} \tag{9}$$

From Eqn. 8 and Eqn. 9, we have:

$$F(w_i^T w_k^{'}) = P_{ik} \tag{10}$$

or

$$w_i^T w_k^{'} = \log(P_{ik}) = \log(X_{ik}) - \log(X_i) \tag{11}$$

From Eqn. 11, we set the following weighed least square objective:

$$J = \sum_{i,j} f(X_{ij})(w_i^T w_j^{'} + b_i + b_j^{'} - \log X_{ij}) \tag{12}$$

where $f(.)$ is a weight function and $b, b^{'}$ are bias terms.

Let's recall the training objective of *skip-gram*:

$$J = -\sum_{i} \sum_{j \in context(i)} \log Q_{ij} \tag{13}$$

where $Q_{ij} = \frac{\exp(w_i^T w_j^{'})}{\sum_k exp(w_i^T w_k^{'})}$, which can be rewritten as:

$$J = -\sum_{i,j} X_{i,j} \log Q_{i,j} = \sum_{i} X_i H(P_i, Q_i) \tag{14}$$

where H is the cross entropy between two distributions.

The *GloVe* model is more advanced than Eqn. 14 in two ways:

(a) The weight function is not X_i but an arbitrary function $f(X_{ij})$.

(b) The cross entropy has several drawbacks. First, it assigns more probability mass to unlikely event. Second, calculating the normalizing constant involves summing over the entire vocabulary, which is costly. To address these problems, the *GloVe* model chooses a quadratic loss instead.

TODO: reread the Pennington's paper.