Neural Word Embeddings from Scratch

Xin Li[1,2]

[1]NLP Center,
Tencent AI Lab

[2]Dept. of System Engineering & Engineering Management,
The Chinese University of Hong Kong

2018-04-09

Outline

What is Word Embedding?

- Word Embedding refers to low dimensional, real-valued dense vectors encoding the semantic information of word.
- Generally, the concepts **Word Embeddings**, **Distributed Word Representations** and **Dense Word Vectors** can be used interchangeably.

John Rupert Firth (linguist)

"You shall know a **word** *by the company it keeps".*

Karl Marx (philosopher)

"The **human essence** *is no abstraction inherent in each single individual. In its reality it is the ensemble of the social relations".*

What is Word Embedding?

Word Embedding is the by-product of neural language model.

- Definition of language model:

$$\mathrm{p}(\mathbf{w}_{1:T}) = \prod_{t=1}^{T} \mathrm{p}(\mathbf{w}_t | \mathbf{w}_{1:t-1})$$

- Neural Language Model (NLM) is the language model where the conditional probability is modeled by neural networks.

Outline

MLP-LM

Bengio et al., JMLR 2003

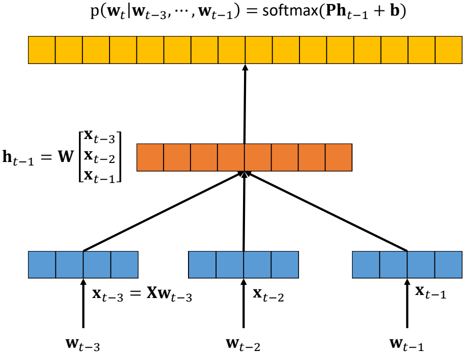


Figure: MLP-LM. n is set to 3.

Training objective is to maximize the log-likelihood:

$$L = \frac{1}{T}\sum \log[p(\mathbf{w}_t|\mathbf{w}_{1:t-n+1})]$$

Conv-LM

Collobert and Weston., ICML 2008

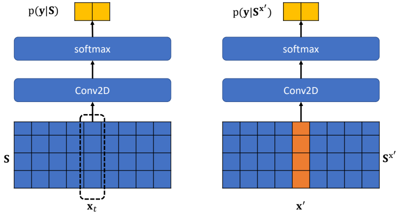


Figure: Conv-LM. $\mathbf{x}_t$ denotes the middle word of $\mathbf{S}$. $\mathbf{S}^{\mathbf{x}'}$ is obtained by replacing middle word of $\mathbf{S}$ with $\mathbf{x}'$

Training objective is to minimize the rank-type loss:

$$L = \sum_{\mathbf{S}\in\mathcal{D}} \sum_{\mathbf{x}'\in\mathcal{V}} \max(0, 1 - \mathrm{p}(\mathbf{y} = 1|\mathbf{S}) + \mathrm{p}(\mathbf{y} = 1|\mathbf{S}^{x'}))$$

RNN-LM

Mikolov et al., INTERSPEECH 2010

Training objective is same to MLP-LM (i.e., maximum likelihood).

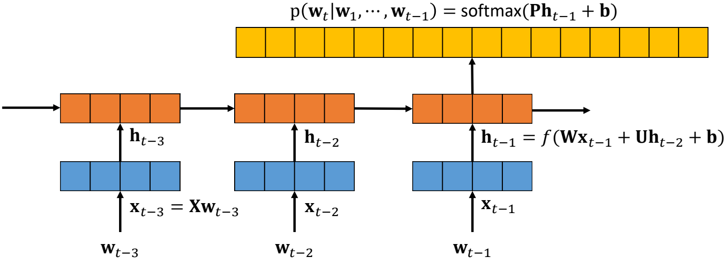


Figure: RNN-LM.

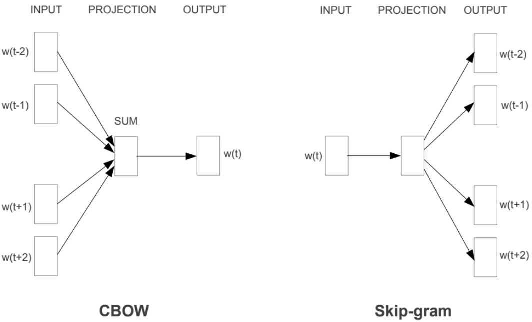
Outline

Word2Vec

(Mikolov et al., ICLR 2013 & NIPS 2013)

Word2Vec involves two different models, namely, **CBOW** and **SG**:

1. **CBOW** (Continuous Bag-of-Words): Using context words to predict the middle word.
2. **SG** (Skip-Gram): Using middle word to predict the context words.



Word2Vec
(Mikolov et al., ICLR 2013 & NIPS 2013)

The main feature of Word2Vec (IMO) is that it is a non-MLE framework, i.e., its aim is not to model the joint probability of the input words.

$$\textbf{CBOW}: \ L = \frac{1}{T}\sum_{t=1}^{T}\log[\mathrm{p}(\mathbf{w}_t|\mathbf{w}_{t-c:t-1}, \mathbf{w}_{t+1:t+c})]$$

$$\textbf{SG}: \ L = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c}\log[\mathrm{p}(\mathbf{w}_{t+j}|\mathbf{w}_t)]$$

Word2Vec is the first model for learning word embeddings from unlabeled data!!!

Word2Vec
(Mikolov et al., ICLR 2013 & NIPS 2013)

Some extensions for improving Word2Vec:

- **HSoftmax** (Hierarchical Softmax)
  1. Full softmax layer is too "fat" since it needs to evaluate $|\mathcal{V}|$ (generally more than 1M in the large corpus) output nodes.
  2. **HSoftmax** takes advantage of binary tree representation of output layer and only needs to evaluate $\log_2(|\mathcal{V}|)$ nodes.

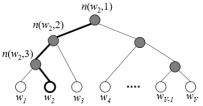


Figure: Binary Tree

word	count
fat	3
fridge	2
zebra	1
potato	3
and	14
in	7
today	4
kangaroo	2

→

Figure: Huffman Tree example

$$p(w_2|w_i) = \prod_{j=1}^{3} \sigma(\mathbb{I}(n(w_2, j+1) = ch(n(w_2, j)))\mathbf{v}_{n(w_2,j)}^{\top}\mathbf{x}_i)$$

Mikolov uses **Huffman Tree** to construct the hierarchical structure.

- **NS** (Negative Sampling) is another alternative for speeding up training.
  1. Formulating $|\mathcal{V}|$-class classification problem as a binary classification problem. ("word prediction" $\Longrightarrow$ "co-occurrence relation prediction")
  2. Training with k additional corrupted samples for each positive sample.

$$L = \log \sigma({\mathbf{x}_O^*}^\top \mathbf{x}_I) + \sum_{i=1}^{k} \mathbb{E}_{w_i^* \sim P_n(w)}[\log \sigma(-{\mathbf{x}_i^*}^\top \mathbf{x}_I)]$$

- **Subsampling**: Most frequent words usually provide less information and randomly discarding them should speedup training and improve performance.

Outline

GloVe

Pennington et al., EMNLP 2014

Motivations of GloVe:

- Global co-occurrence count is the primary information for generating word embeddings. (Word2Vec ignores this kind of information)
- Only using co-occurrence information is not enough to distinguish relevant words from irrelevant words.

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

The appropriate starting point for learning word vectors should be ratios of co-occurrence probabilities!!!

GloVe

Pennington et al., EMNLP 2014

$$
\begin{aligned}
&F(\mathbf{x}_i, \mathbf{x}_j, \hat{\mathbf{x}}_k) = \mathrm{p}_{ik}/\mathrm{p}_{jk} \\
\Longrightarrow &F(\mathbf{x}_i - \mathbf{x}_j, \hat{\mathbf{x}}_k) = \mathrm{p}_{ik}/\mathrm{p}_{jk} \\
\Longrightarrow &F((\mathbf{x}_i - \mathbf{x}_j)^\top \hat{\mathbf{x}}_k) = \mathrm{p}_{ik}/\mathrm{p}_{jk}
\end{aligned} \tag{1}
$$

The roles of word and context word should be exchangeable, thus:

$$
F((\mathbf{x}_i - \mathbf{x}_j)^\top \hat{\mathbf{x}}_k) = F(\mathbf{x}_i^\top \hat{\mathbf{x}}_k)/F(\mathbf{x}_j^\top \hat{\mathbf{x}}_k) \tag{2}
$$

$$
\begin{aligned}
\text{According to (1) and (2): } F(\mathbf{x}_i^\top \hat{\mathbf{x}}_k) = \mathrm{p}_{ik} &\Longrightarrow \mathbf{x}_i^\top \hat{\mathbf{x}}_k = \log(C_{ik}) - \log(C_i) \\
&\Longrightarrow \mathbf{x}_i^\top \hat{\mathbf{x}}_k + b_i + \hat{b}_k = \log(C_{ik})
\end{aligned} \tag{3}
$$

$$
\text{Training objective: } J = \sum_{i,k=1}^{|\mathcal{V}|} f(C_{ik})(\hat{\mathbf{x}}_k + b_i + \hat{b}_k - \log(C_{ik}))^2 \tag{4}
$$

where $f(C_{ik})$ is a weight function to filter the noise from rare co-occurrences.

Outline

SG-NS as Implicit Matrix Factorization

Levy and Goldberg, NIPS 2014 & TACL 2015

- Skip-Gram with Negative Sampling (SG-NS) can be efficiently trained and achieve state-of-the-art results.
- The outputs of SG-NS are word embeddings $\mathbf{X}^w$ and context word embeddings $\mathbf{X}^c$ (ignored).

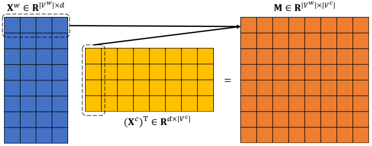


Figure: SG-NS

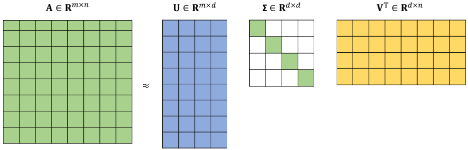


Figure: SVD for d-rank factorization

SG-NS as Implicit Matrix Factorization

Levy and Goldberg, NIPS 2014 & TACL 2015

Training objective of SG-NS:

$$\ell(I, O) = \log \sigma({\mathbf{x}_O^*}^\top \mathbf{x}_I) + \sum_{i=1}^{k} \mathbb{E}_{w_i^* \sim P_n(w)}[\log \sigma(-{\mathbf{x}_i^*}^\top \mathbf{x}_I)]$$

$$L = \sum_{I \in \mathcal{V}^w} \sum_{O \in \mathcal{V}^c} C_{IO} \ell(I, O)$$

The optimal value is attained at:

$$y = {\mathbf{x}_O^*}^\top \mathbf{x}_I = \log(\frac{C_{IO} * |\mathcal{D}|}{C_I * C_O}) - \log k$$

The first item is the point-wise mutual information (PMI) of word pair (I, O).

SG-NS as Implicit Matrix Factorization

Levy and Goldberg, NIPS 2014 & TACL 2015

- The matrix $\mathbf{M}^{\mathrm{PMI}_k}$, called "*shifted* PMI Matrix", emerges as the optimal solution for SG-NS's objective. Each cell of the matrix is defined below:

$$\mathbf{M}_{ij}^{\mathrm{PMI}_k} = \mathbf{X}_i^w \cdot \mathbf{X}_j^c = \mathbf{x}_i \cdot \mathbf{x}_j^* = \mathrm{PMI}(i,j) - \log k$$

- The objective of SG-NS can be regarded as a weighted matrix factorization problem over $\mathbf{M}^{\mathrm{PMI}_k}$.
- The matrices $\mathbf{M}_0^{\mathrm{PMI}_k}$ and $\mathbf{M}^{\mathrm{PPMI}_k}$ can be better alternatives of $\mathbf{M}^{\mathrm{PMI}_k}$.

$$(\mathbf{M}_0^{\mathrm{PMI}_k})_{ij} = \begin{cases} 0 & \text{if } C_{ij} = 0 \\ \mathbf{M}_{ij}^{\mathrm{PMI}_k} & \text{Otherwise} \end{cases}$$

$$\mathbf{M}_{ij}^{\mathrm{PPMI}_k} = \mathrm{PPMI}(i,j) - \log k$$
$$\mathrm{PPMI}(i,j) = \max(\mathrm{PMI}(\mathrm{i},\ \mathrm{j}), 0)$$

Outline

SVD over shifted PPMI matrix

Levy and Goldberg, NIPS 2014 & TACL 2015

- *shifted* PPMI matrix:

$$\mathbf{M}^{\mathrm{SPPMI}_k} = \mathrm{SPPMI}_k(i,j), \text{ where } \mathrm{SPPMI}_k(i,j) = \max(\mathrm{PMI}(i,j) - \log k, 0)$$

- Performing SVD over $\mathbf{M}^{\mathrm{SPPMI}_k}$ and $\mathbf{U} \cdot \sqrt{\mathbf{\Sigma}}$ is treated as word representations $\mathbf{X}^w$.
  - This method outperforms SG-NS on word similarity task!!!

Outline

Outline