Algorithm for the skip gram word2vec model

9

let $\mathbf{U} \in \mathbb{R}^{K \times N}$ be our center word embeddings corresponding to $\mathbf{x}_w$

$\forall(\mathbf{x}_w, \mathbf{x}_c) \in D$ do:

$$\mathbf{U}_w \leftarrow \mathbf{U}_w - \eta \nabla_{\mathbf{U}_w} NLL$$

set $\mathbf{u}_w = \mathbf{U}_w$ via the non-zero indice of our one-hot center word $\mathbf{x}_w$

cross entropy loss: $\mathbf{L}(U, V \mid \mathbf{x}_w, \mathbf{x}_c) = -\mathbf{x}_c \cdot \log p(\mathbf{x}_c \mid \mathbf{x}_w\ ; \mathbf{U}, \mathbf{V})$

gradient descent:

$O(NK)$

let $\mathbf{V} \in \mathbb{R}^{K \times N}$ be our center word embeddings corresponding to $\mathbf{x}_c$

compute inner product between $\mathbf{u}_w$ and all context vectors $\mathbf{V} : \mathbf{u}_w \cdot \mathbf{V} \in \mathbb{R}^N$

$$\text{compute probability over all context words given center word} : \frac{\exp(\mathbf{u}_w \cdot \mathbf{V})}{\sum_{j=1}^{N} \exp(\mathbf{u}_w \cdot \mathbf{V})_j}$$

gradients: $\nabla_{U_w} NLL = \mathbf{V} \cdot \left(P_{\mathbf{x}_c|\mathbf{x}_w} - \mathbf{x}_c \right)^T \in \mathbb{R}^K$

$$\nabla_V NLL = \mathbf{u}_w \cdot \left(P_{\mathbf{x}_c|\mathbf{x}_w} - \mathbf{x}_c \right) \in \mathbb{R}^{K \times N}$$

$$\mathbf{V} \leftarrow \mathbf{V} - \eta \nabla_{\mathbf{V}} NLL$$

encodes
distributional
hypothesis

only w^{th} row of $\mathbf{U}$ gets updated entire $\mathbf{V}$ gets updated

Algorithm for the skip gram word2vec model

let $\mathbf{U} \in \mathbb{R}^{K \times N}$ be our center word embeddings corresponding to $\mathbf{x}_w$

let $\mathbf{V} \in \mathbb{R}^{K \times N}$ be our center word embeddings corresponding to $\mathbf{x}_c$

$\forall (\mathbf{x}_w, \mathbf{x}_c) \in D$ do:

set $\mathbf{u}_w = \mathbf{U}_w$ via the non-zero indice of our one-hot center word $\mathbf{x}_w$

compute inner product between $\mathbf{u}_w$ and all context vectors $\mathbf{V}$: $\mathbf{u}_w \cdot \mathbf{V} \in \mathbb{R}^N$

encodes distributional hypothesis

compute probability over all context words given center word : $\dfrac{\exp(\mathbf{u}_w \cdot \mathbf{V})}{\sum_{j=1}^{N} \exp(\mathbf{u}_w \cdot \mathbf{V})_j}$

$O(NK)$

cross entropy loss: $\mathbf{L}(U, V \mid \mathbf{x}_w, \mathbf{x}_c) = -\, \mathbf{x}_c \cdot \log p(\mathbf{x}_c \mid \mathbf{x}_w \,; \mathbf{U}, \mathbf{V})$

gradients: $\nabla_{U_w} NLL = \mathbf{V} \cdot \left(P_{\mathbf{x}_c|\mathbf{x}_w} - \mathbf{x}_c\right)^T \in \mathbb{R}^K$

$\nabla_V NLL = \mathbf{u}_w \cdot \left(P_{\mathbf{x}_c|\mathbf{x}_w} - \mathbf{x}_c\right) \in \mathbb{R}^{K \times N}$

gradient descent: $\mathbf{U}_w \leftarrow \mathbf{U}_w - \eta \nabla_{\mathbf{U}_w} NLL$ only w^{th} row of $\mathbf{U}$ gets updated

$\mathbf{V} \leftarrow \mathbf{V} - \eta \nabla_{\mathbf{V}} NLL$ entire $\mathbf{V}$ gets updated

For large N, we need to avoid the partition function

- The approach on the previous slide is the preferred method when N is a manageable size, say $N < 10^5$.

- When $N > 10^6$, the partition function (denominator of the softmax function) becomes prohibitively expensive to compute due the O(NK) scaling.

- There are several approaches to get around having to compute the (full) partition function:
    - Hierarchical softmax
    - Importance sampling (IS)
    - Adaptive IS
    - Target sampling
    - Noise contrastive estimation (NCE)
    - Negative sampling

$\sim 10 - 100$x speedup