Natural Language Processing

Word Embedding

1. Introduction

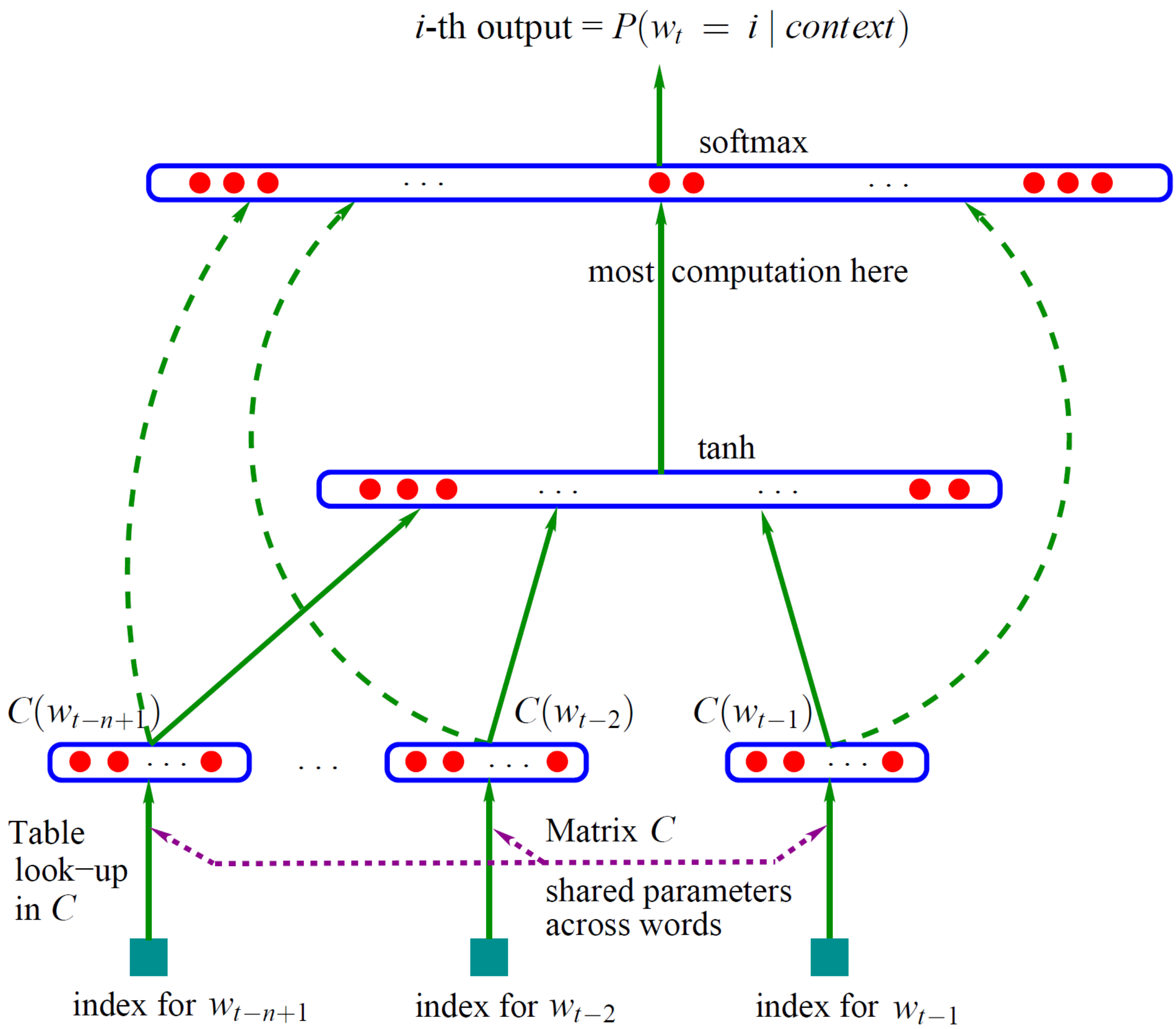
One-hot vector: Represent every word as an $\mathbb{R}^{|V|\times 1}$ vector with all 0s and one 1 at the index of that word in the sorted english language. For example:

$$w^{aardvark} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{at} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots, w^{zebra} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

But,

$$(w^{hotel})^T w^{motel} = (w^{hotel})^T w^{cat} = 0$$

2. Neural Probabilistic Language Model (Bengio et al. NIPS 2001)



input: $w_{t-n+1}, \ldots, w_{t-2}, w_{t-1}$

look-up: $C(w_{t-n+1}), \ldots, C(w_{t-2}), C(w_{t-1})$

first layer: $x = (C(w_{t-n+1}), \ldots, C(w_{t-2}), C(w_{t-1}))$

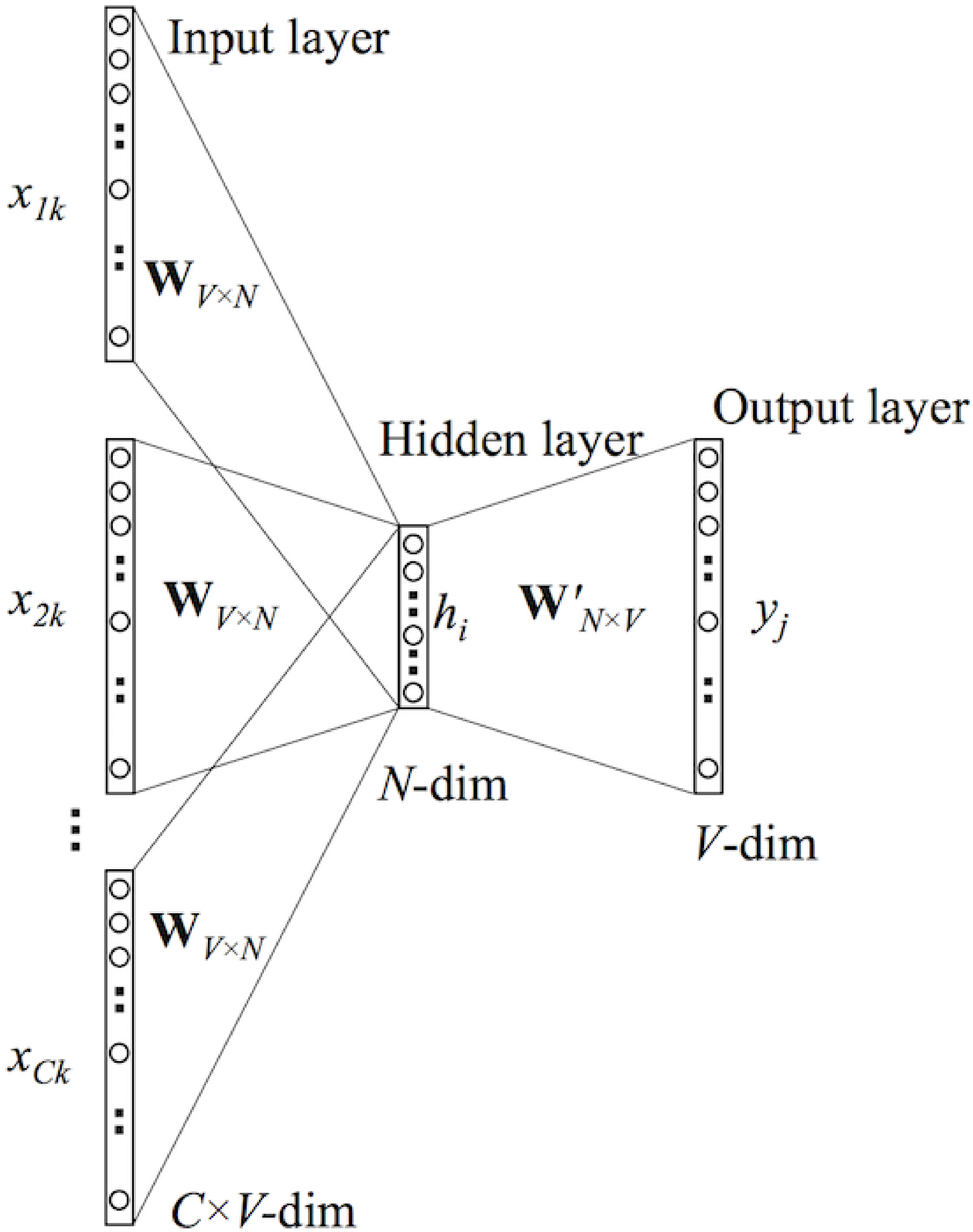
hidden layer: $h = \tanh(d + Hx)$

output layer: $y = \text{softmax}(b + Wx + U\tanh(d + Hx))$

loss function: $L(w_t, y) = -w_t \log(y)$

3. Continuous Bag of Words Model (Mikolov et al. ICLR 2013)

CBOW Model: Predicting a center word form the surrounding context.



one-hot word vectors: $x^{(i-C)}, \ldots, x^{(i-1)}, x^{(i+1)}, \ldots, x^{(i+C)}$

embedding:
$u^{(i-C)} = W^{(1)}x^{(i-C)}, u^{(i-C+1)} = W^{(1)}x^{(i-C+1)}, \ldots, u^{(i+C)} = W^{(1)}x^{(i+C)}$

hidden layer: $h = \frac{u^{(i-C)} + u^{(i-C+1)} + \cdots + u^{(i+C)}}{2C}$

output layer: $\hat{y} = \mathrm{softmax}(z) = \mathrm{softmax}(W^{(2)}h)$

loss function: $H(\hat{y}, y) = -y\log(\hat{y}) = -\sum_{j=1}^{|V|} y_j \log(\hat{y}_j)$

optimization(SGD):

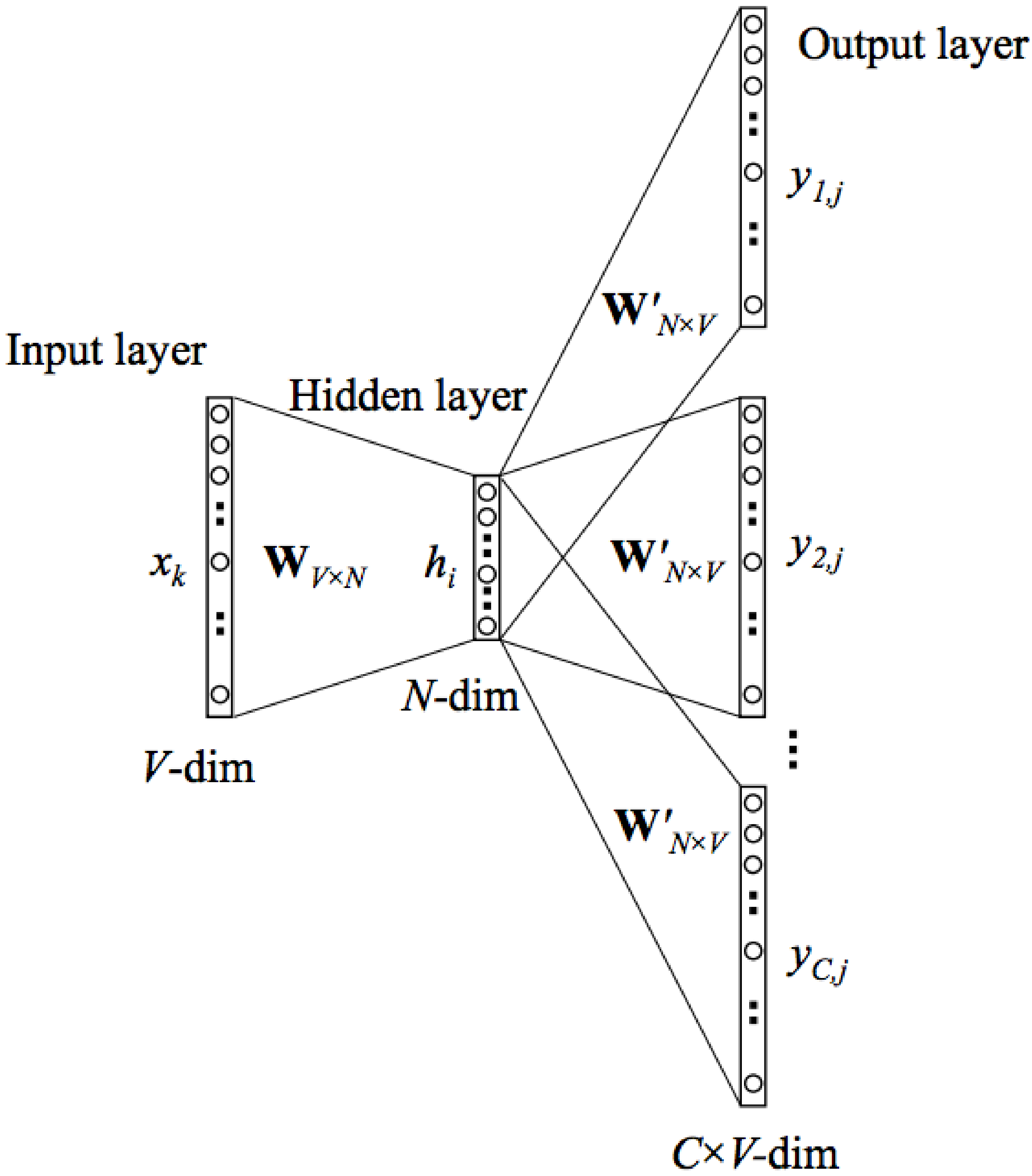
$$
\begin{aligned}
\frac{\partial H}{\partial W^{(1)}} &= \frac{\partial H}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W^{(1)}} = \frac{\partial H}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial W^{(1)}} \\
&= \frac{\partial H}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial h} \cdot \frac{\partial h}{\partial W^{(1)}} \\
\frac{\partial H}{\partial \hat{y}} &= -\frac{y}{\hat{y}} \\
\frac{\partial \hat{y}}{\partial z} : \frac{\partial \hat{y}_i}{\partial z_i} &= \hat{y}_i(1-\hat{y}_i), \frac{\partial \hat{y}_i}{\partial z_j} = -\hat{y}_i \cdot \hat{y}_j \\
\frac{\partial z}{\partial h} &= W^{(2)} \\
\frac{\partial h}{\partial W^{(1)}} &= \frac{1}{2C} \sum_k \frac{\partial u^{(k)}}{\partial W^{(1)}}
\end{aligned}
$$

$$
\frac{\partial H}{\partial W^{(1)}} = (\hat{y} - y)W^{(2)} \otimes \frac{1}{2C} \sum_k x^{(k)}
$$

$$
\begin{aligned}
\frac{\partial H}{\partial W^{(2)}} &= \frac{\partial H}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W^{(2)}} = \frac{\partial H}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial W^{(2)}} \\
&= (\hat{y} - y) \cdot \frac{\partial z}{\partial W^{(2)}} \\
&= (\hat{y} - y) \otimes h
\end{aligned}
$$

4. Skip-Gram Model (Mikolov et al. ICLR 2013)

Skip-Gram Model: Predicting surrounding context words given a center word.



one-hot input vector: x

embedding: $u = W^{(1)}x$

hidden layer: $h = u$

output layer: $y = \text{softmax}(v) = \text{softmax}(W^{(2)}h)$

loss function: $H(\hat{y}, Y) = -\sum_i y_i \log(\hat{y})$

Reference

1. **Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. Journal of Machine Learning Research (JMLR), 3:1137–1155, 2003.** http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf
2. **Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In Proceedings of Workshop at ICLR, 2013.** http://arxiv.org/pdf/1301.3781v3.pdf
3. **word2vec:** https://code.google.com/p/word2vec/
4. **Deep Learning in NLP（一）词向量和语言模型**: http://licstar.net/archives/328
5. **Recurrent Neural Network Language Models**: http://www.rnnlm.org/