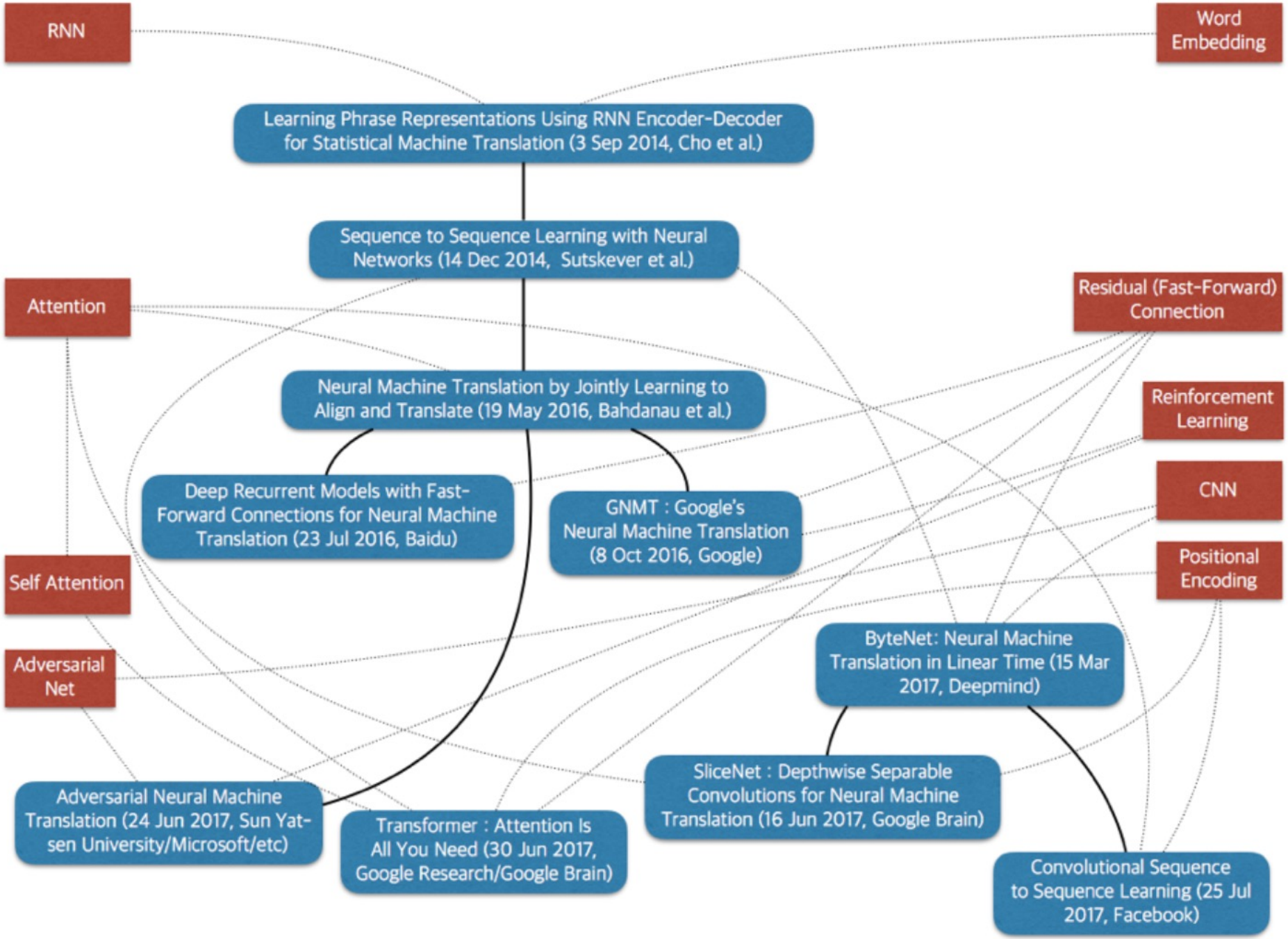
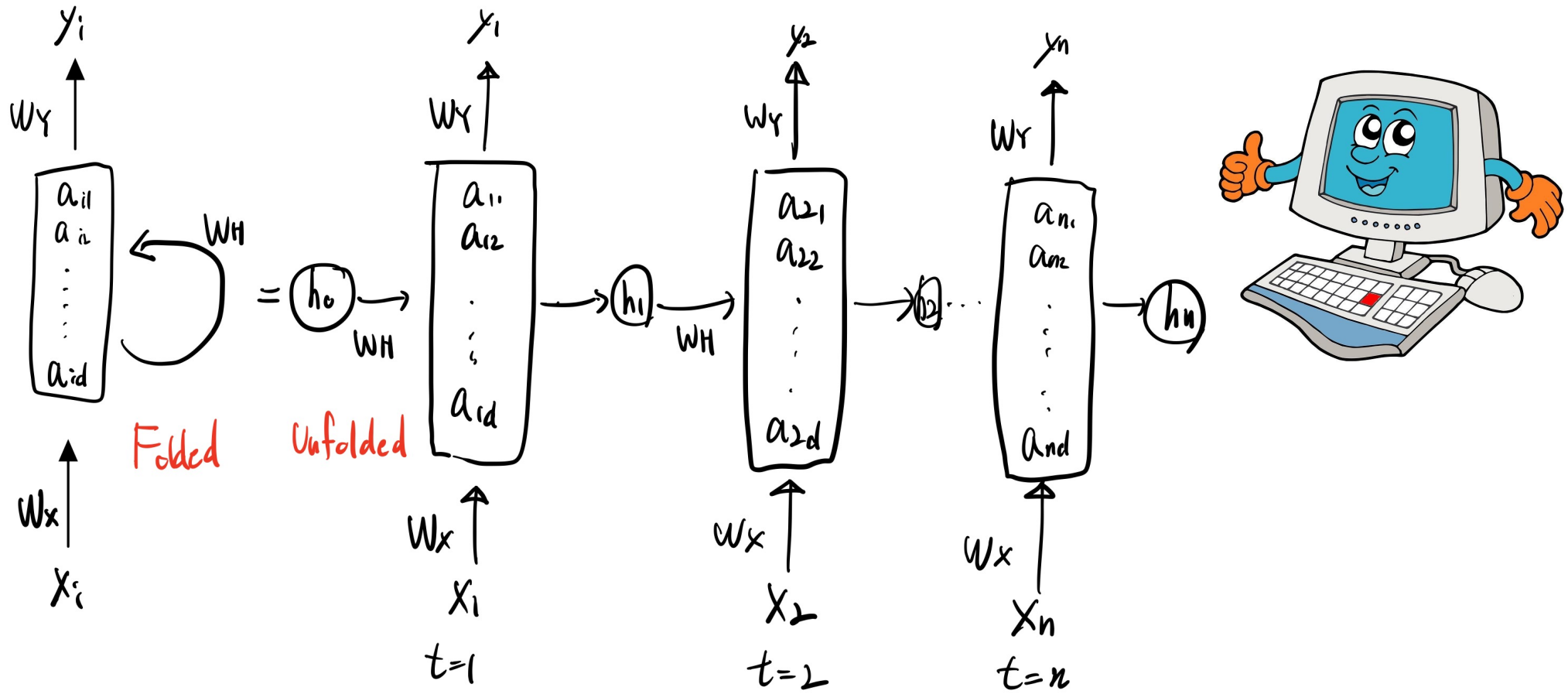
Embedding:
How to Vectorize Text Data

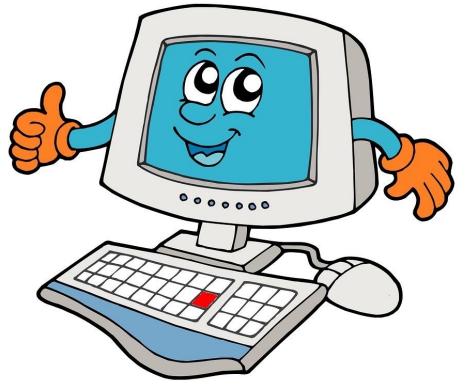

RNN
Word Embedding
Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation (3 Sep 2014, Cho et al.)
Sequence to Sequence Learning with Neural Networks (14 Dec 2014, Sutskever et al.)
Attention
Residual (Fast-Forward) Connection
Neural Machine Translation by Jointly Learning to Align and Translate (19 May 2016, Bahdanau et al.)
Reinforcement Learning
Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation (23 Jul 2016, Baidu)
GNMT : Google's Neural Machine Translation (8 Oct 2016, Google)
CNN
Self Attention
Positional Encoding
ByteNet: Neural Machine Translation in Linear Time (15 Mar 2017, Deepmind)
Adversarial Net
SliceNet : Depthwise Separable Convolutions for Neural Machine Translation (16 Jun 2017, Google Brain)
Adversarial Neural Machine Translation (24 Jun 2017, Sun Yat-sen University/Microsoft/etc)
Transformer : Attention Is All You Need (30 Jun 2017, Google Research/Google Brain)
Convolutional Sequence to Sequence Learning (25 Jul 2017, Facebook)

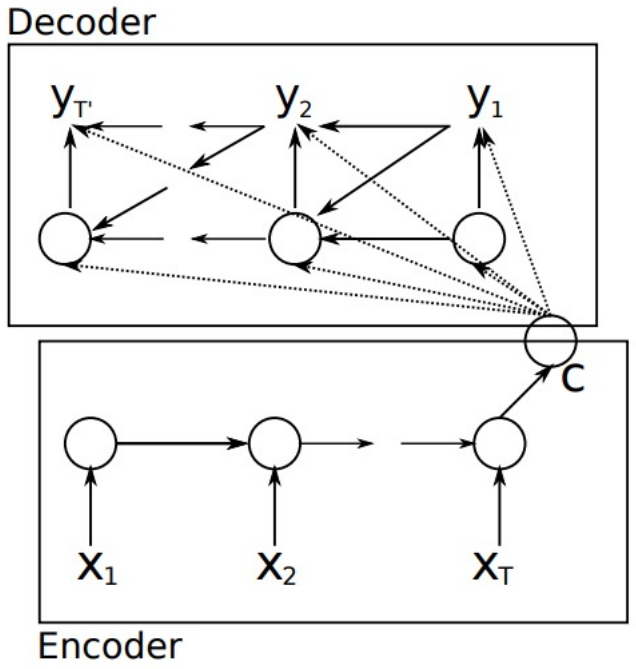
Reminder



RNN Encoder-Decoder

nice to meet you



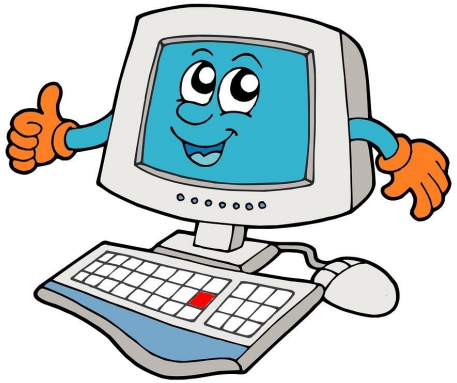


만나서 반갑습니다

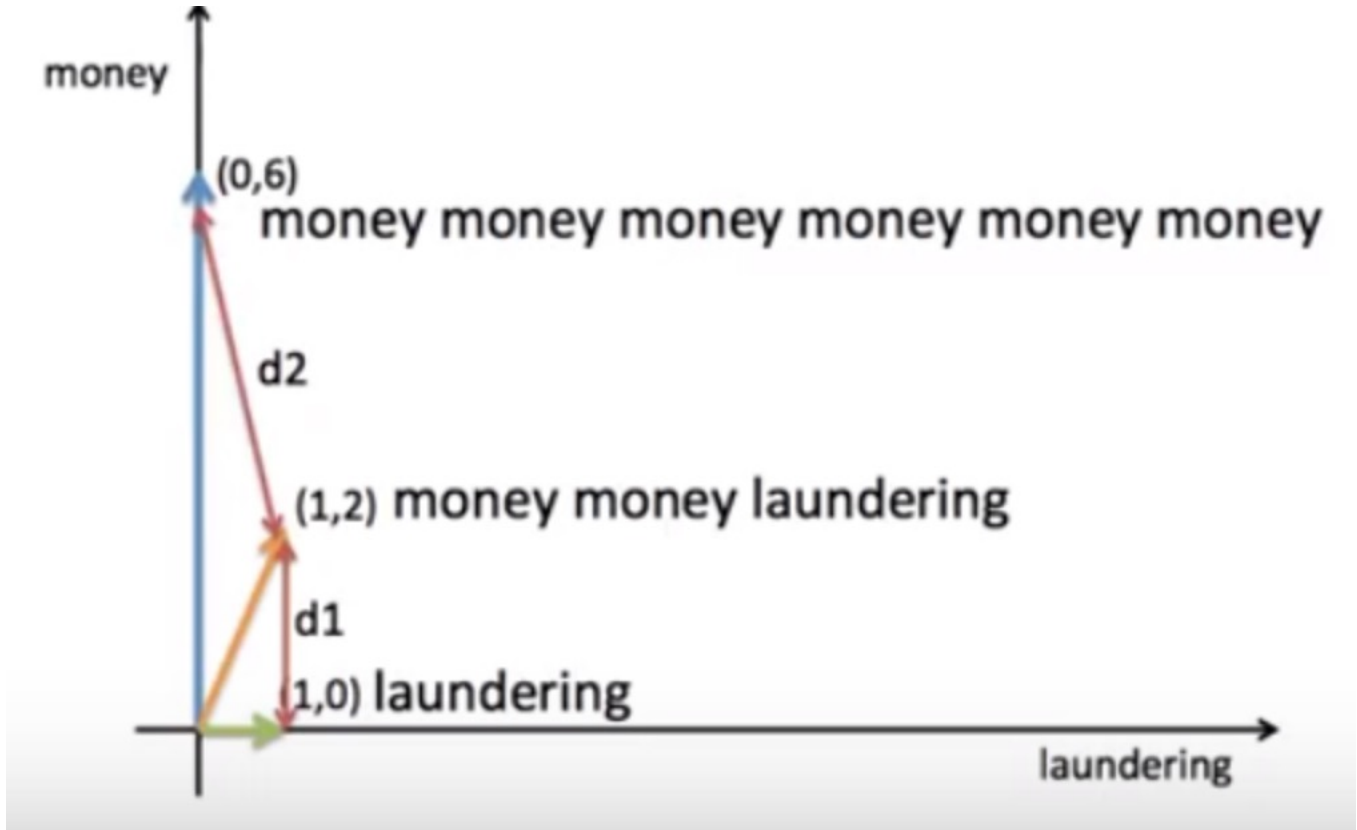
one-hot vector

너구리 : [1, 0, 0]
라쿤 : [0, 1, 0]
사람: [0, 0, 1]

1. (데이터가 늘어날수록) 아주 커짐
2. 내적하면 모두 0 (서로 독립)



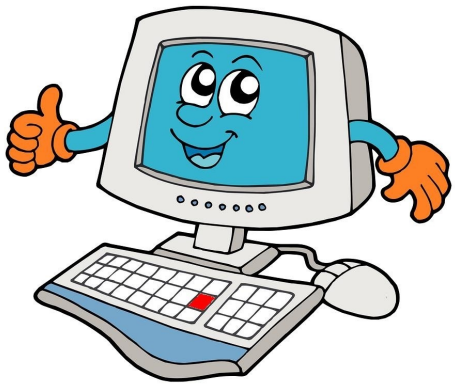
one-hot vector



one-hot vector

너구리 : [1, 0, 0]
라쿤 : [0, 1, 0]
사람: [0, 0, 1]

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$



코사인 유사도 : -1

코사인 유사도 : 0

코사인 유사도 : 1

코드 리뷰

NNLM (Neural Network Language Model)

language model: 주어진 단어로 다음 단어를 예측하는 모델

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

statistical model of language:
language can be represented by the conditional probability of the next word given all the previous ones.

$$\hat{P}(w_1^T) = \prod_{t=1}^{T} \hat{P}(w_t | w_1^{t-1}),$$

$$w_i^j = (w_i, w_{i+1}, \cdots, w_{j-1}, w_j)$$

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

fact: temporally closer words in the word sequence are statistically more dependent
-> n-gram models construct tables of conditional probabilities for the next word

$$\hat{P}(w_t|w_1^{t-1}) \approx \hat{P}(w_t|w_{t-n+1}^{t-1}).$$

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

1. 각각의 단어를 분산된 word feature vector 로 본다.
2. 시퀀스의 feature vector를 시퀀스의 joint probability function으로 표현한다.
3. 각각의 word feature vector와 probability function의 파라미터를 동시에 학습시킨다.

그렇다면 확률 함수를 이전 단어들이 주어졌을 때 이후 단어의 conditional probability로 표현 가능함.
이를 학습 데이터들의 Log-likelihood를 최대화하는 식으로 학습 가능.

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

$w_1 \cdots w_T$ of words $w_t \in V$ training set, where the vocabulary V is a large but finite set.

$f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t | w_1^{t-1})$ objective: learn a good model

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

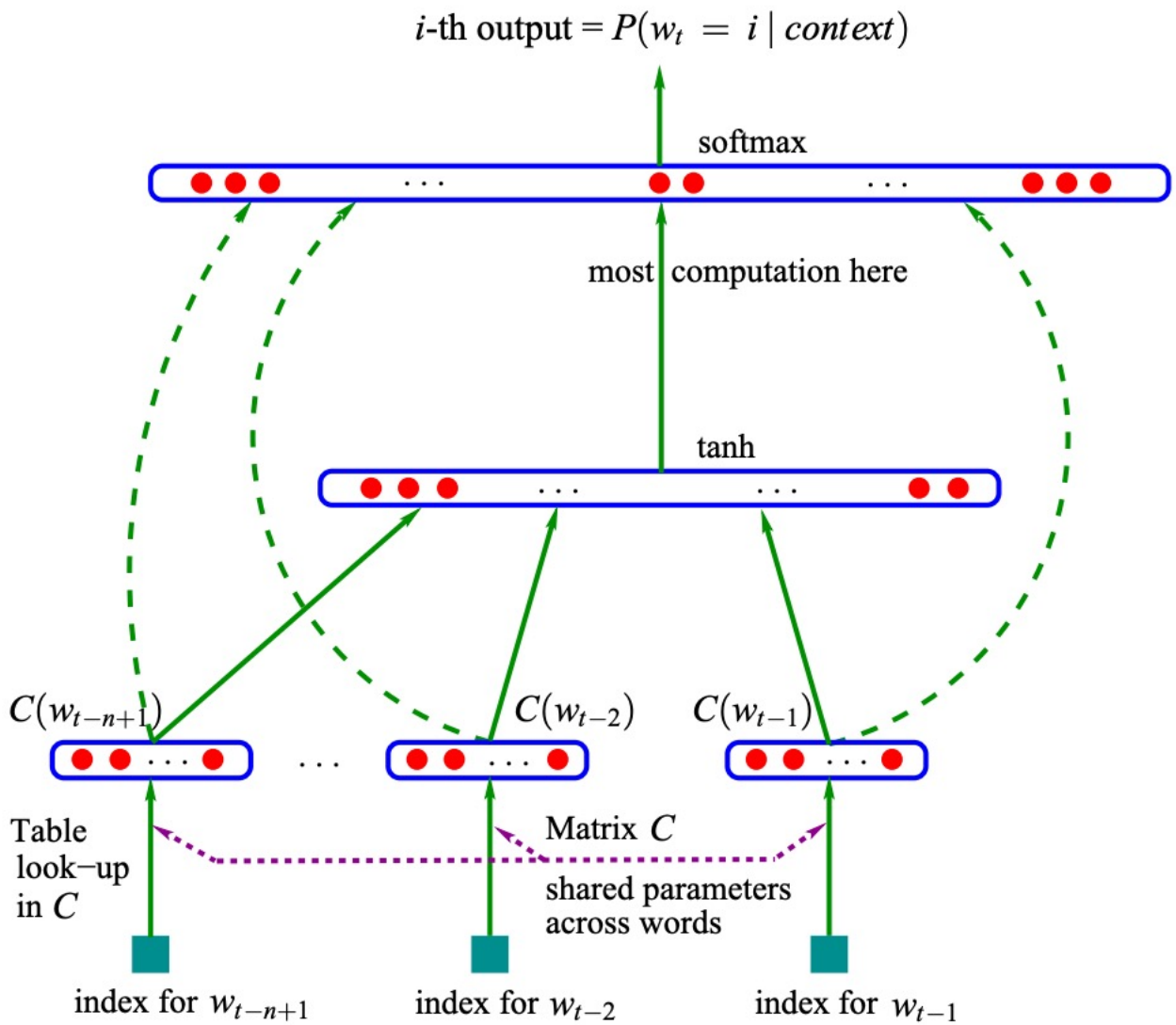
We decompose the function $f(w_t, \cdots, w_{t-n+1}) = \hat{P}(w_t | w_1^{t-1})$ in two parts:

1. A mapping C from any element i of V to a real vector $C(i) \in \mathbb{R}^m$. It represents the *distributed feature vectors* associated with each word in the vocabulary. In practice, C is represented by a $|V| \times m$ matrix of free parameters.

2. The probability function over words, expressed with C: a function g maps an input sequence of feature vectors for words in context, $(C(w_{t-n+1}), \cdots, C(w_{t-1}))$, to a conditional probability distribution over words in V for the next word w_t. The output of g is a vector whose i-th element estimates the probability $\hat{P}(w_t = i | w_1^{t-1})$ as in Figure 1.

$$f(i, w_{t-1}, \cdots, w_{t-n+1}) = g(i, C(w_{t-1}), \cdots, C(w_{t-n+1}))$$

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)



A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

$$L = \frac{1}{T}\sum_t \log f(w_t, w_{t-1}, \cdots, w_{t-n+1}; \theta) + R(\theta),$$

$$\hat{P}(w_t | w_{t-1}, \cdots w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}}.$$

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

$x_t = C \cdot w_t$

when $V = 3$.

V: 말뭉치의 단어수

m: x_t의 차원수

$$\underset{m \times 3}{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & & \\ \vdots & & \\ a_{m1} & \cdots & a_{m3} \end{pmatrix}} \underset{3 \times (n-1)}{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}} \Rightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

C

w_t

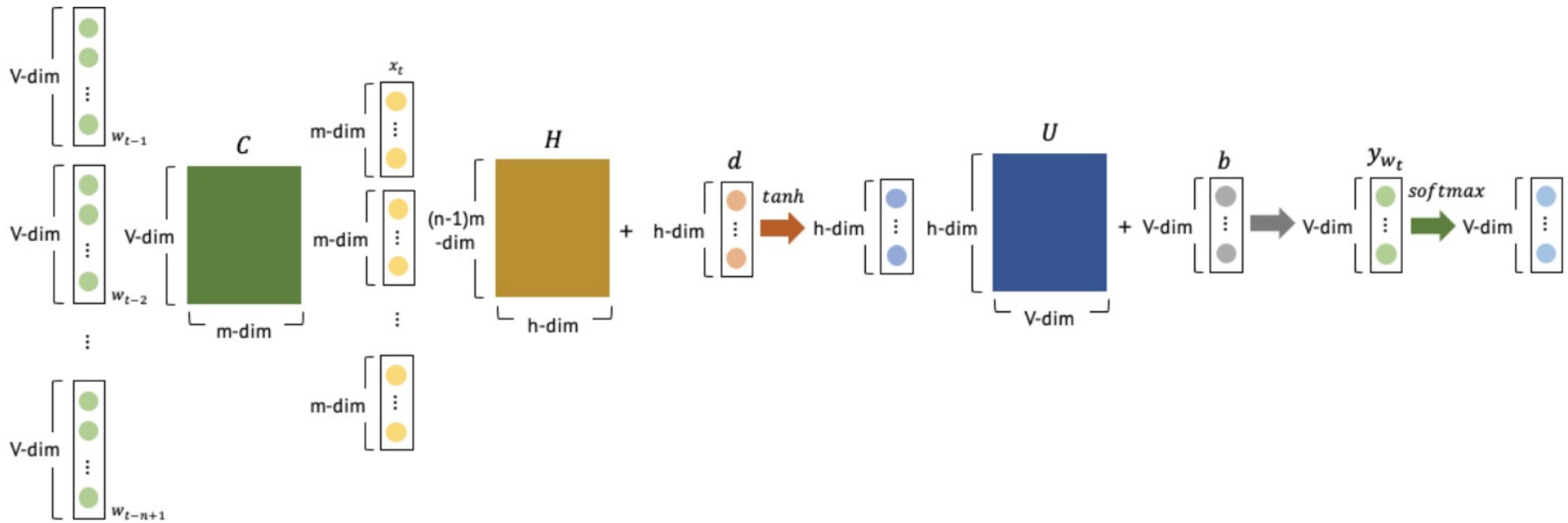
y_{w_t}

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

$$y_{w_t} = b + U \cdot tanh(d + Hx_t)$$

$$H \in R^{h\times(n-1)m}, \quad x_t \in R^{(n-1)\times m}, \quad d \in R^{h\times 1}$$
$$U \in R^{|V|\times h}, \quad b \in R^{|V|}, \quad y \in R^{|V|} \quad , C \in R^{m\times|V|}$$

A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)



A neural probabilistic language model. Journal of Machine Learning Research (Bengio et al, 2003)

장점

- 저장 공간 이점
- 단어의 유사도 표현

단점

- 많은 파라미터 training → 과적합 문제, 계산 비용
- n개만의 단어 참고

Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

computational complexity per each training example

$$Q = N \times D + N \times D \times H + H \times V$$

projection layer expensive!

N: previous word → one-hot encoding
V : 말뭉치의 단어 수
P: N X D shared projection matrix

we try to minimize computational complexity.

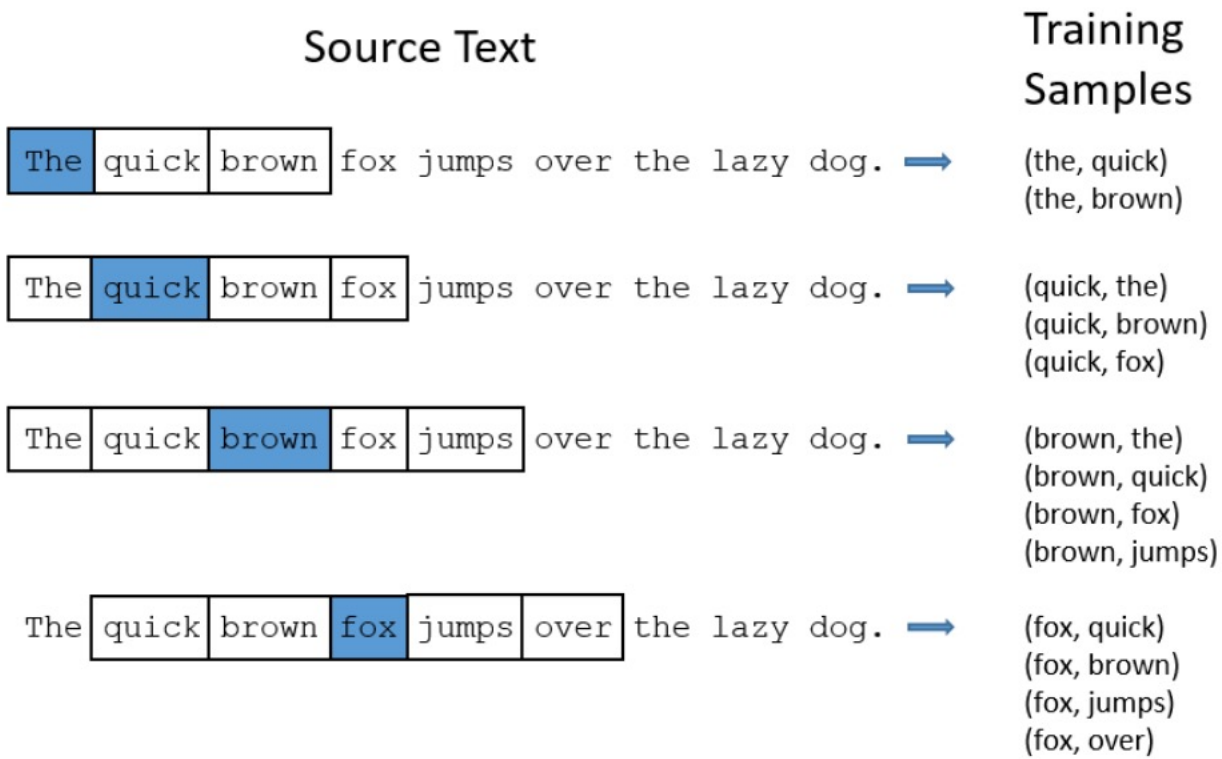
Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

CBOW (Continuous Bag-of-Words Model)

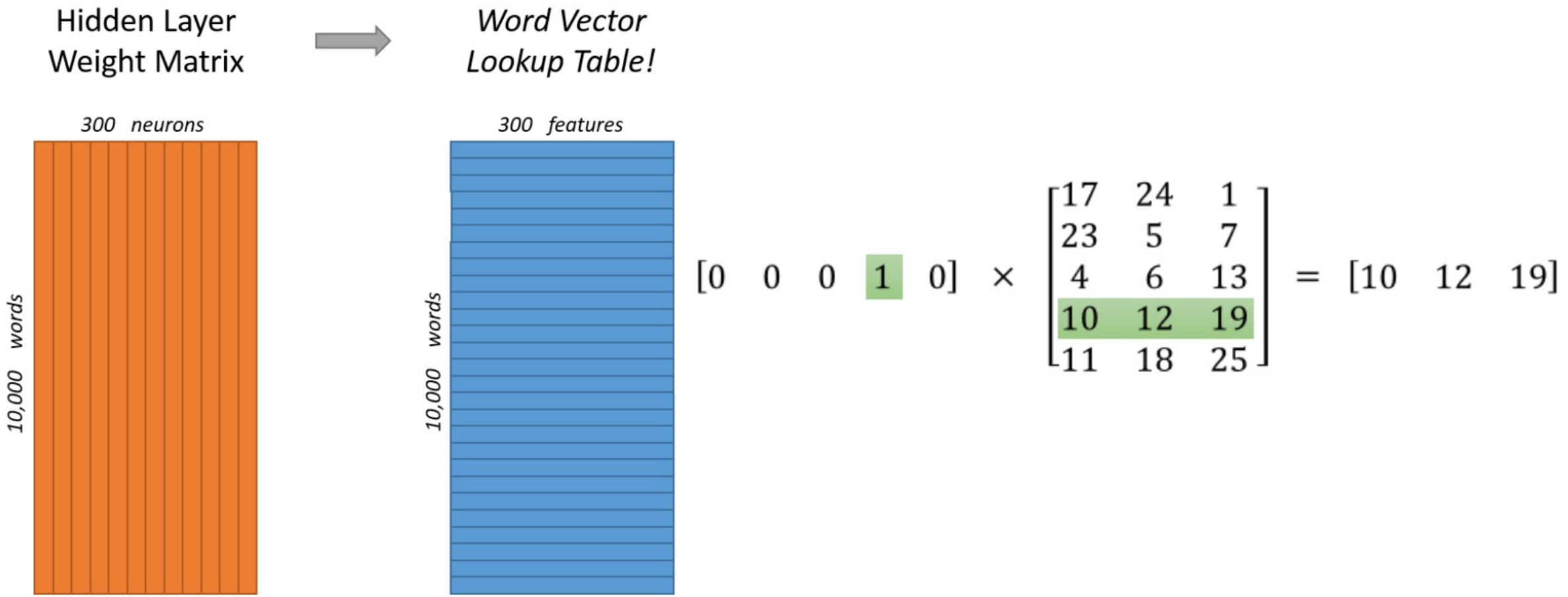
1. remove non-linear hidden layer
2. projection layer is shared for all words (order of words does not influence)
3. use words from the future

$$Q = N \times D + D \times log_2(V).$$

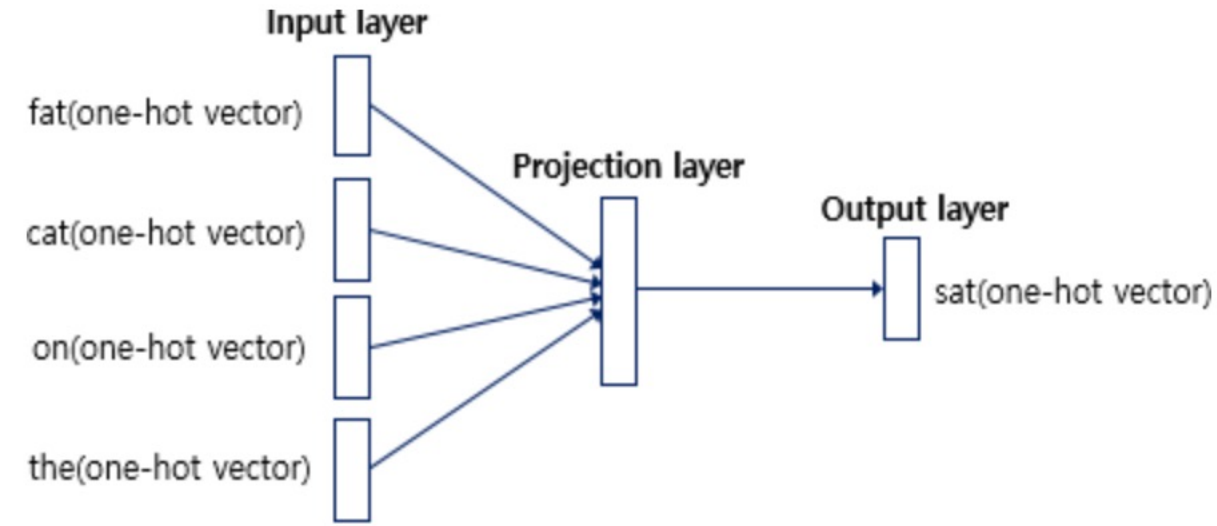
Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

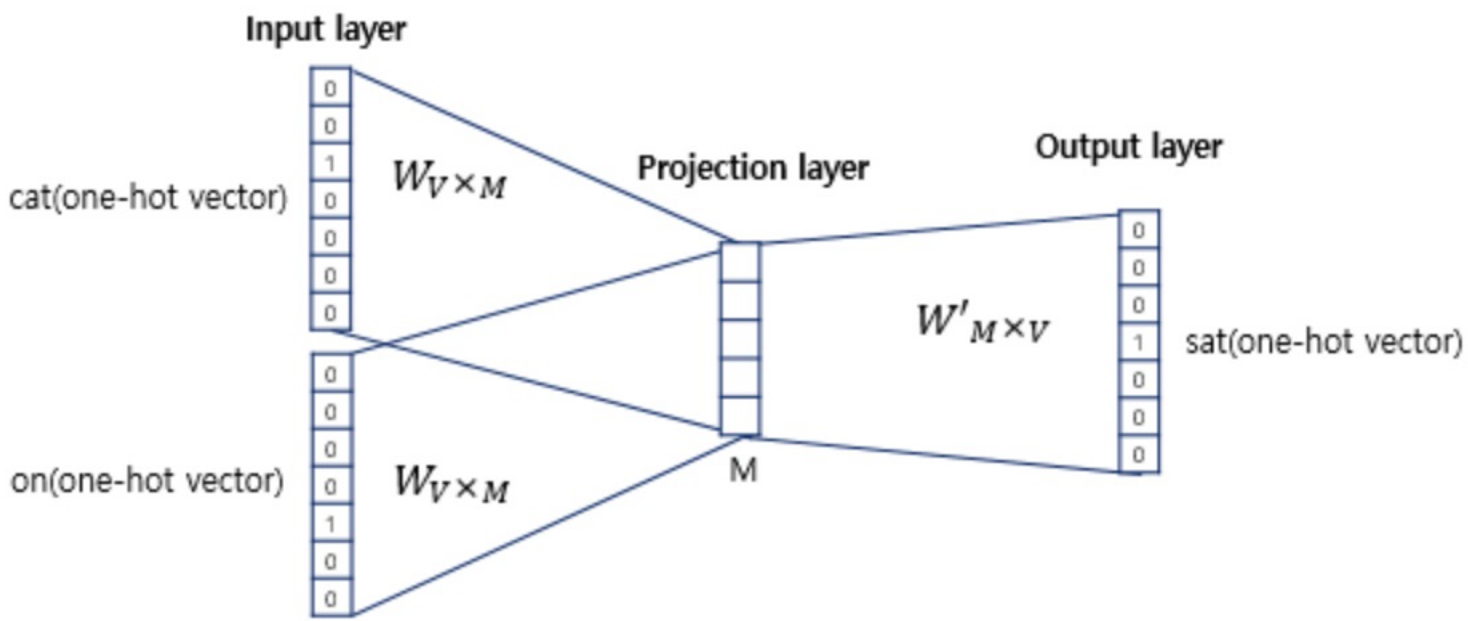


Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

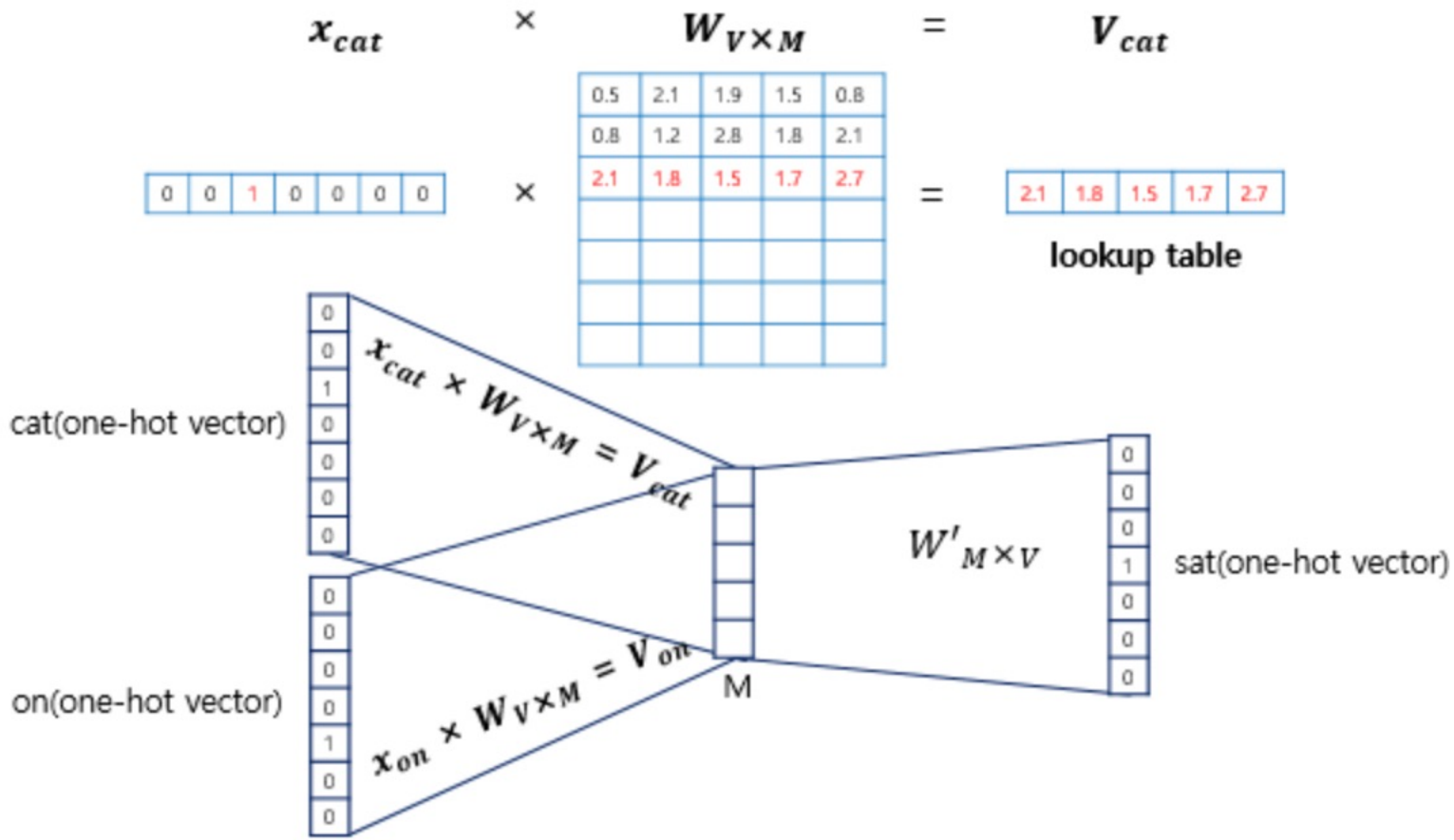


Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

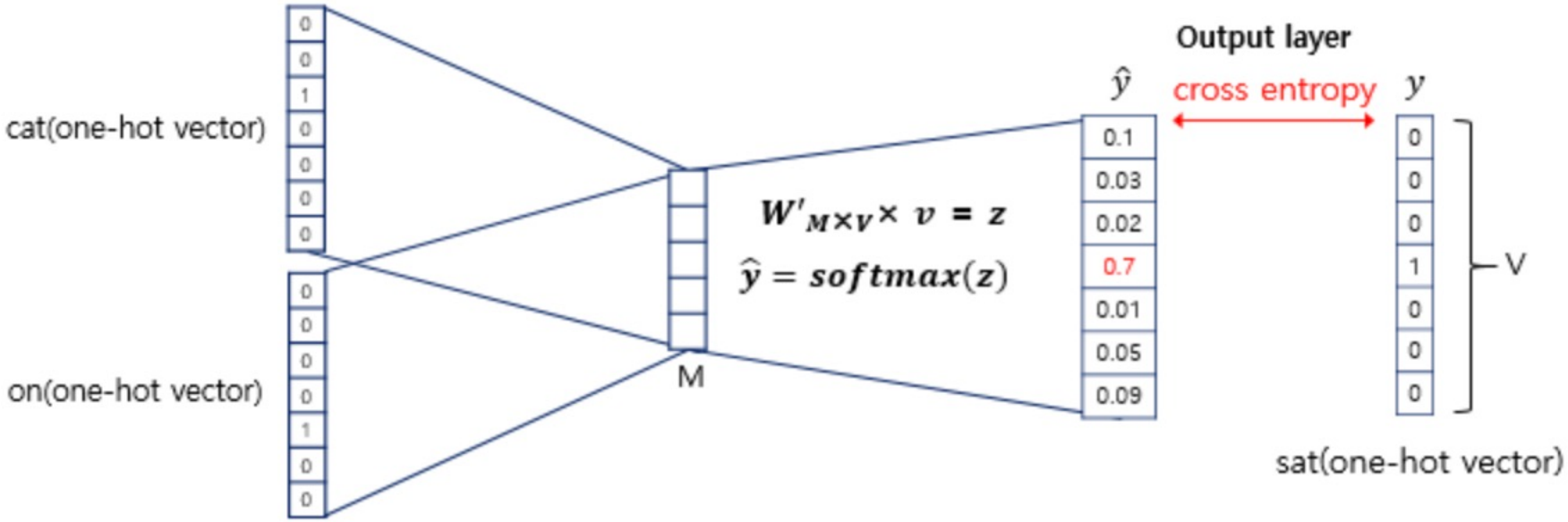




Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)



Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)



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

Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

$$
\begin{aligned}
minimize J &= -\log P(w_c | w_{c-m}, \ldots, w_{c+m}) \\
&= -\log P(u_c | v) \\
&= -\log \frac{exp(u_c^\intercal \hat{v})}{\sum_{j=1}^{|V|} exp(u_j^\intercal \hat{v})} \\
&= -u_c^{intercal} \hat{v} + \log \sum_{j=1}^{|V|} exp(u_j^\intercal \hat{v})
\end{aligned}
$$

Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

$$P(o|c) = \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)}$$

$$\begin{aligned}
\frac{\partial}{\partial v_c} \ln P(o|c) &= \frac{\partial}{\partial v_c} \ln \frac{exp(u_o^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)} \\
&= \frac{\partial}{\partial v_c} u_o^T v_c - \frac{\partial}{\partial v_c} \ln \sum_{w=1}^{W} exp(u_w^T v_c) \\
&= u_o^T - \frac{1}{\sum_{w=1}^{W} exp(u_w^T v_c)} (\sum_{w=1}^{W} exp(u_w^T v_c) \cdot u_w) \\
&= u_o^T - \sum_{w=1}^{W} \frac{exp(u_w^T v_c)}{\sum_{w=1}^{W} exp(u_w^T v_c)} \cdot u_w \\
&= u_o^T - \sum_{w=1}^{W} P(w|c) \cdot u_w
\end{aligned}$$

$$v_c^{t+1} = v_c^t + \alpha(u_o^T - \sum_{w=1}^{W} P(w|c) \cdot u_w)$$

Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

subsampling : I 번째 단어를 학습에서 제외 (계산량 감소)

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$$

Effficient Estimation of Word Representations in Vector Space (Mikolov et al. 2013)

negative sampling : 전체 단어를 구하지 않고, 일부 단어를 계산

$$P(w_i) = \frac{f(w_i)^{3/4}}{\sum_{j=0}^{n} f(w_j)^{3/4}}$$

superhero dataset에서 superman과 history가 가장 비슷한 히어로 10명 찾기

https://www.kaggle.com/jonathanbesomi/superheroes-nlp-dataset