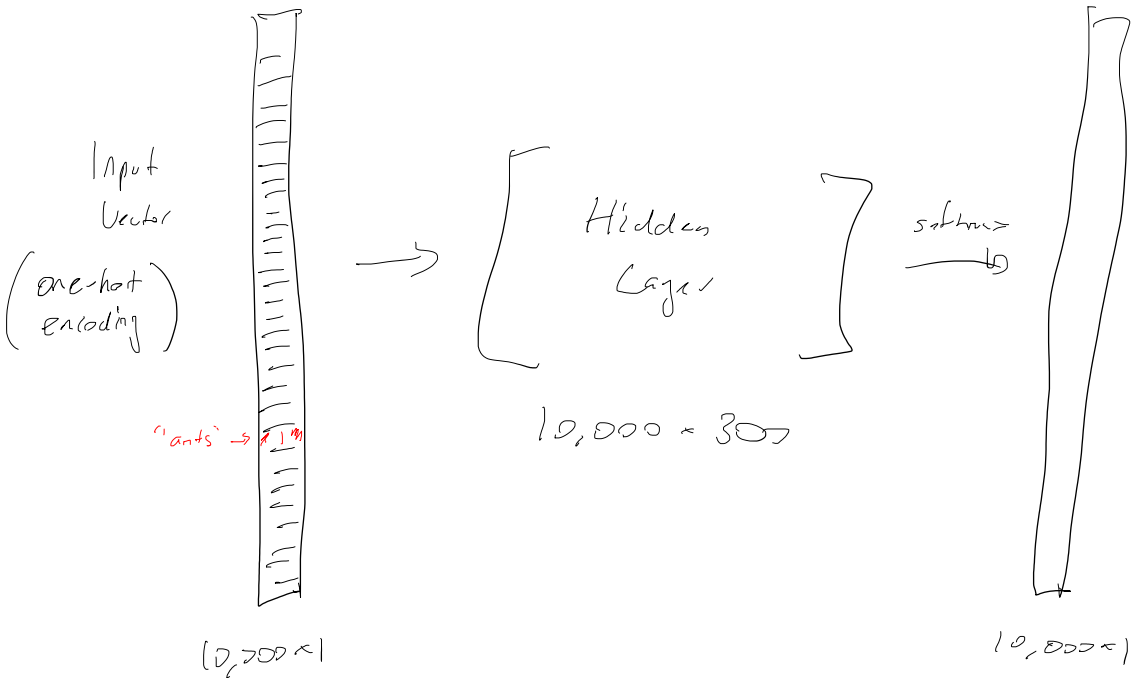
Word 2 Vec

Skip-gram

- Similar trick as using encoder/decoder for unsupervised learning + strip away output layer, use just hidden layer

- Give model a "fake task", i.e. given a word, estimate the probability of being a context word, for each word in the vocabulary
  - i.e. word is within a window



$$[0 \quad 0 \quad 0 \quad 1 \quad 0] \times \begin{bmatrix} 17 & 24 & 1 \\ 23 & 5 & 7 \\ 4 & 6 & 13 \\ 10 & 12 & 19 \\ 11 & 18 & 25 \end{bmatrix} = [10 \quad 12 \quad 19]$$

Softmax classifier

Softmax: $\frac{e^{\theta x}}{\sum e^{\theta x}}$

So: Input is a $1 \times 10{,}000$ one-hot vector.

Hidden layer is $10{,}000 \times 300$ hidden layer.

The product is a 1×300 vector.

This is input into a softmax classifier

(So each output neuron has a weight vector, θ_i
then calculates $\exp(\theta_i x)$
↑ output, 1×300 vector

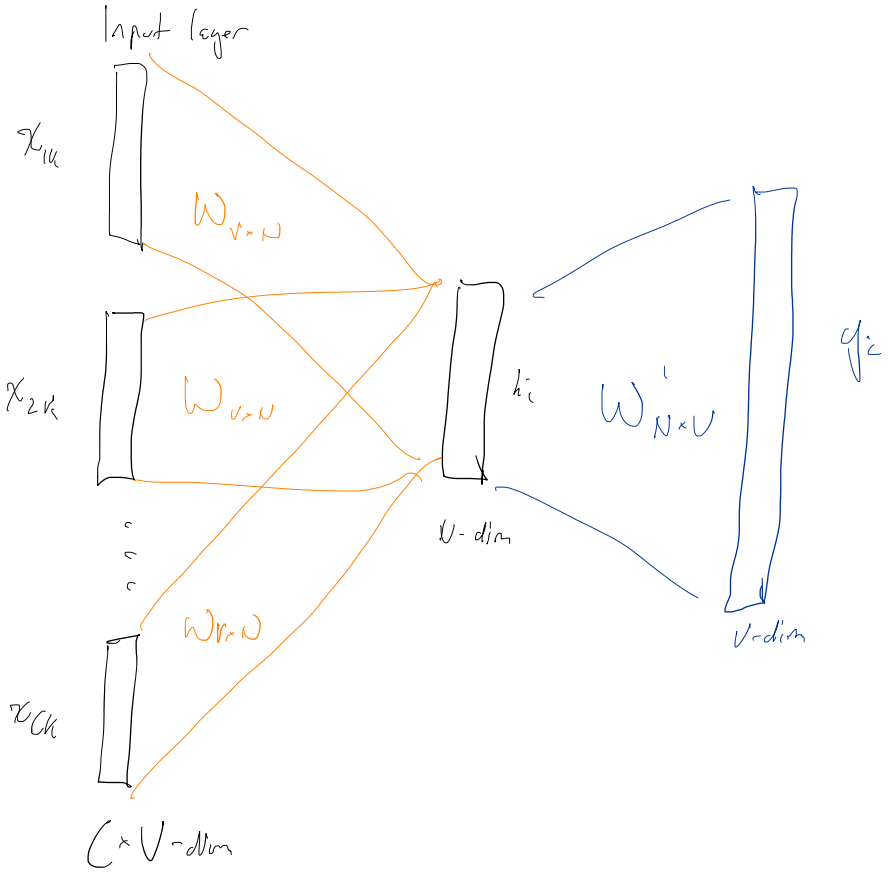
then to normalize, $\frac{\exp(\theta_i x)}{\sum \exp(\theta_i x)}$

CBOW (Continuous bag of words)

- Predict center word from "context"

the cat jumped over the fence.

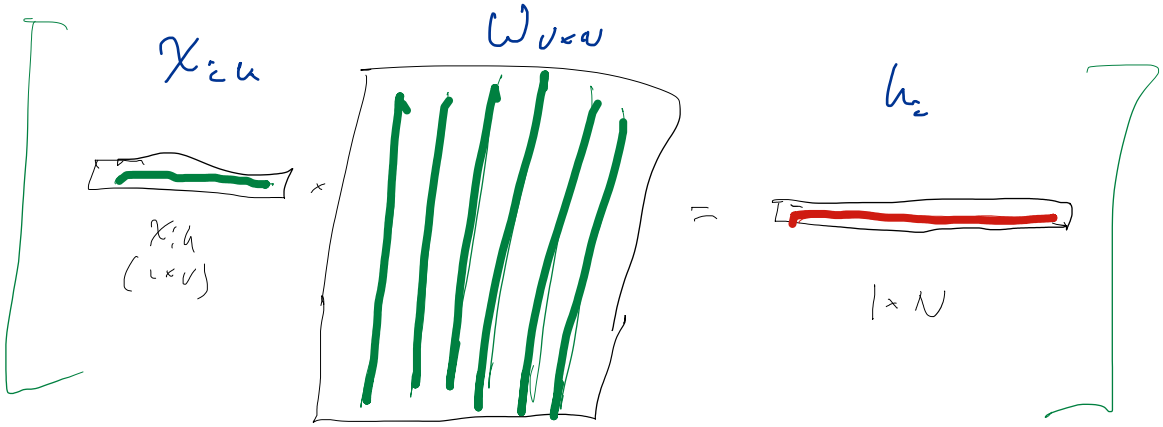
- Let
  - C: window size
  - V: vocabulary size
  - h: hidden layer, N-dimensional vector (N is arbitrary)
  - W: $V \times N$ weight matrix
  - W': $N \times V$ weight matrix
- (skip-gram model, reversed)



1. $\{x_{1k}, x_{2k}, \ldots, x_{Ck}\}$ are one-hot encoded vectors

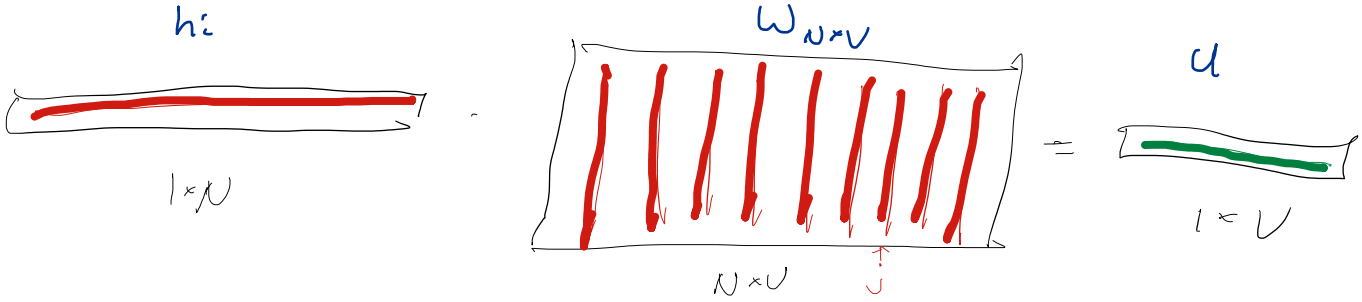
2. Concatenated to form $C \times V$-dim matrix

3. Multiply with weights $W_{V\times N}$ to get $C \times N$ matrix, which is averaged to get a $1 \times N$ vector.



$$h = \frac{1}{C} W \cdot \left(\sum_{i=1}^{C} x_i\right)$$

(4) Multiply the hidden layer vector $(1 \times N)$ with W' $(N \times V)$



Remember:

$$(B_0 \quad B_1)\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = (a_{11}B_0 + a_{21}B_1, \; a_{12}B_1 + a_{22}B_1)$$

So $u_j = v'_{w_j} \cdot h$

- h: hidden layer
- v'_{w_j}: j^{th} column from matrix $W'_{N \times V}$
- pass u_j through softmax

$$y_j = p(w_{y_j} \mid w_1, \ldots, w_C) = \frac{\exp(u_j)}{\sum_{j=1}^{V} \exp(u'_j)}$$

Softmax

Softmax:

e.g.

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 2.0 \\ 1.0 \\ 0.1 \end{bmatrix}$$

$$\frac{e^{y_1}}{\sum_{j=1}^{3} e^{y_j}} = \frac{e^2}{e^2 + e^1 + e^{0.1}} \approx 0.7$$