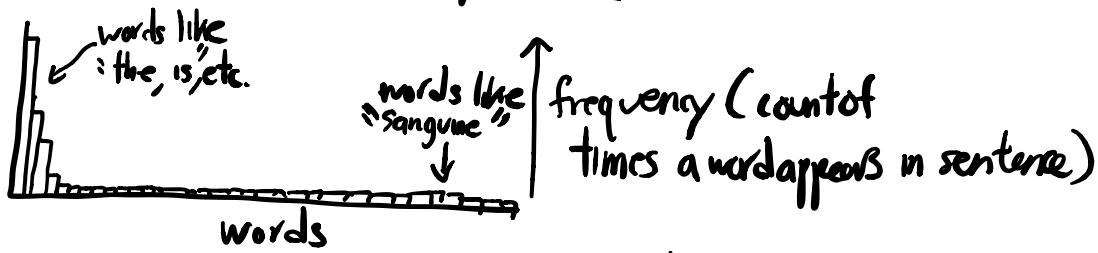


## Problems with count vectorizers:

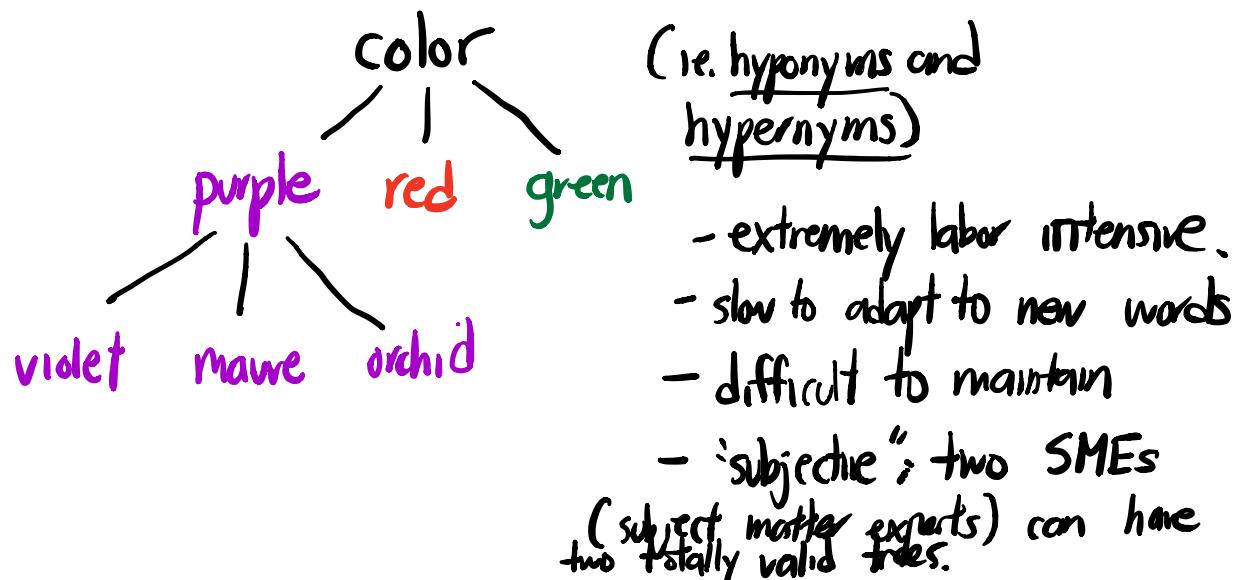
- ① each word is equidistant in its vector representation:

intuitively, we → dog: [ 0 0 1 ]  
 know dog should  
 be more similar  
 to canine than  
 banana.  
 canine: [ 0 1 0 ]  
 banana: [ 1 0 0 ] } all dot products are  
 the same (0).  
 The angles made by a pair  
 of vectors are all orthogonal:

- ② counts skew towards high frequency stopwords (the, is, on, I, like)  
 Remember the long tail distribution of words we saw in Week 1 (Zipf's Law):



## Problems with rule-based domain trees:



So ... what do we do?

We make a new assumption: the distributional hypothesis: "You shall know a word by the company it keeps" - JR Firth, (British linguist).

### Continuous Bag of Words

I used class notes to study for the test.

(+3    +2    +1    +    +1    +2    +3)

context window ( $m=3$ )

$$P(x_t = \text{study} | x_{t-3} = \text{class}) \times$$

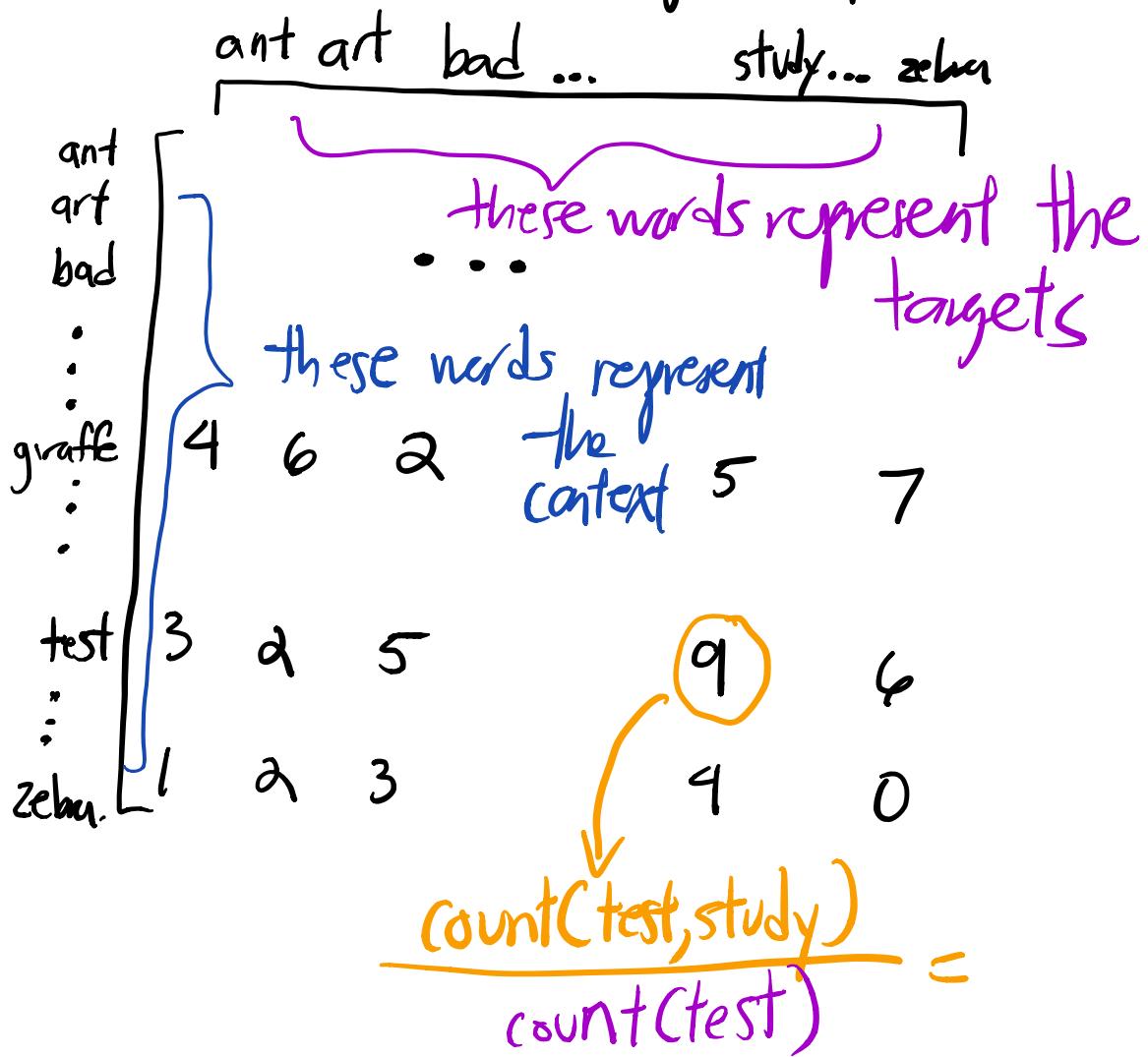
$$P(x_t = \text{study} | x_{t-2} = \text{notes}) \times$$

...

$$\boxed{P(x_t = \text{study} | x_{t+3} = \text{test})}$$

probability that study is the target word given that the context word is test.

How would we find this probability?



$$\frac{9}{3+2+5+9+6} = \frac{9}{25} = .36$$

However, in practice, we typically do not use CBOW, but rather skipgram, which is the opposite of CBOW:

	<u>Inputs</u>	<u>Target</u>
CBOW	context	target word
skipgram	target word	context

$$P(X_{t+3} = \text{class} | X_t = \text{study})$$

$$P(X_{t+1} = t_0 | \dots | X_t = \text{study}) \times \dots \Rightarrow \prod_{\substack{-M \leq m \leq M \\ m \neq 0}} P(X_{t+m} | X_t)$$

$$P(X_{t+3} = \text{test} | X_t = \text{study})$$

"for each context word in the window"

$$M=3$$

context window

size is 3.

We do this for each target

word : for each target word

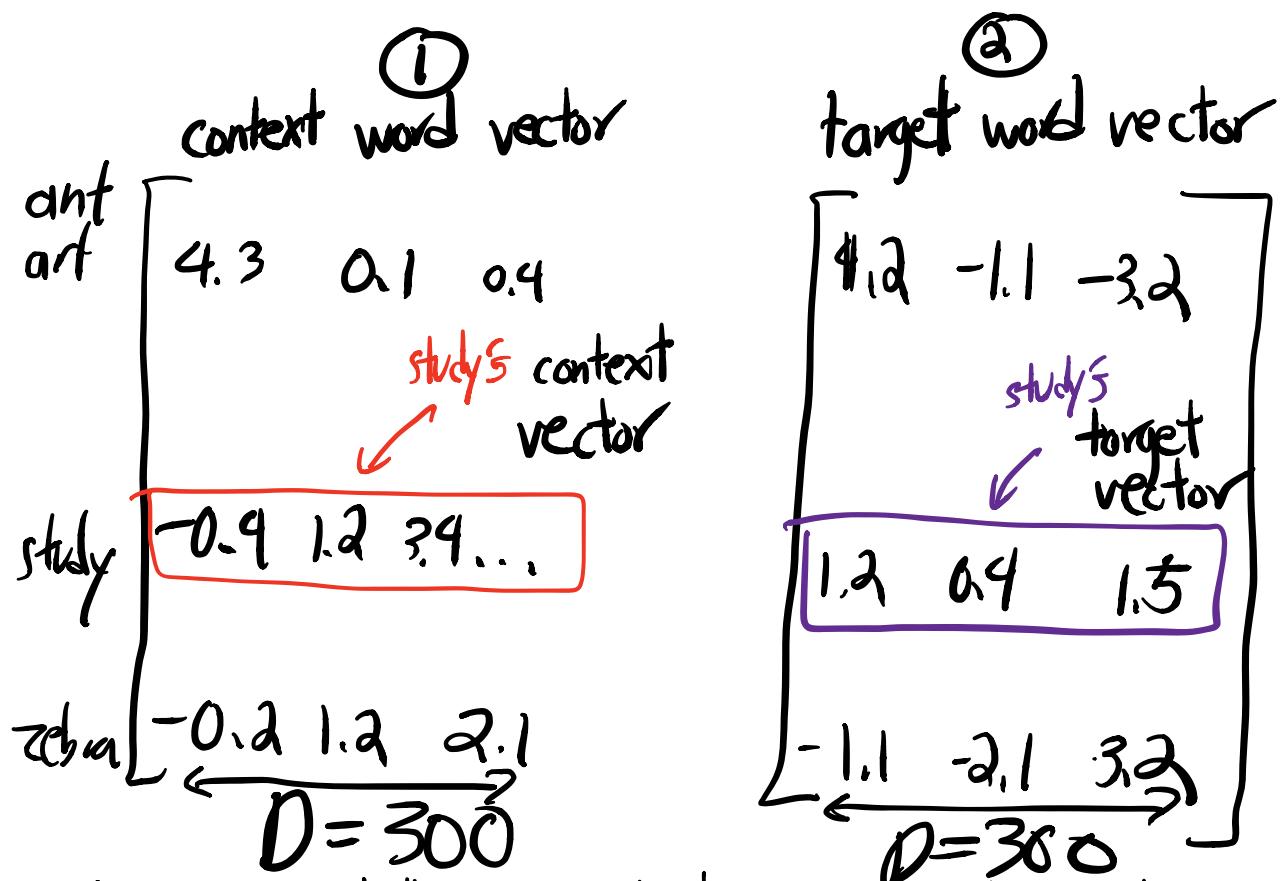
$$\prod_{t=1}^T \prod_{\substack{-M \leq m \leq M \\ m \neq 0}} P(X_{t+m} | X_t, \theta)$$

↓                      ↑                      ↗  
 context word      target word      weights  
 (our parameters)

This becomes our objective function that we optimize (the higher this value the more "correct" our model is):

$$J(\theta) = \prod_{t=1}^T \prod_{\substack{-M \leq m \leq M \\ m \neq 0}} P(X_{t+m} | X_t, \theta)$$

So what is  $\theta$ ? It is 2 matrices:



Each word has both a context vector and a target vector.

Therefore, in skipgram, for each target word, we typically have  $2x$  window size context words to predict on:

this  
is  
fed  
into  
the  
model

$\left\{ \begin{array}{l} (\text{study}, \text{to}) \\ (\text{study}, \text{like}) \\ (\text{study}, \text{mattered}) \end{array} \right.$

these are our  
 $y$  variables

Our objective function:

① maximize probability:

$$J(\theta) = \prod_{t=1}^T \prod_{-m \leq j \leq m, j \neq 0} p(w_{t+j} | w_t, \theta)$$

② convert to negative log-likelihood:

$$\text{NLL } J(\theta) = -\frac{1}{T} \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{t+j} | w_t, \theta)$$

$$\text{NLL } J(\theta) = \frac{1}{T} \sum_{t=1}^T \sum_{\substack{-m \leq j \leq m \\ j \neq 0}} \log p(w_{t+j} | w_t, \theta)$$

we converted  $w_j \mapsto z_j$  so we end up with this log inside.

probability of finding the word  $w_{t+j}$  in the same context for each context word of the word  $w_t$

take the average over all  $T$  words in corpus

for each word in the corpus

We attempt to minimize the NLL  $J(\theta)$  our objective function.

How do we find  $p(w_{t+1} | w_t, \theta)$ ?

$W_t$  = study (or target word)

$W_{t+j}$  = like (our context word)

$$\Theta = \{W, C\}$$

our word matrix

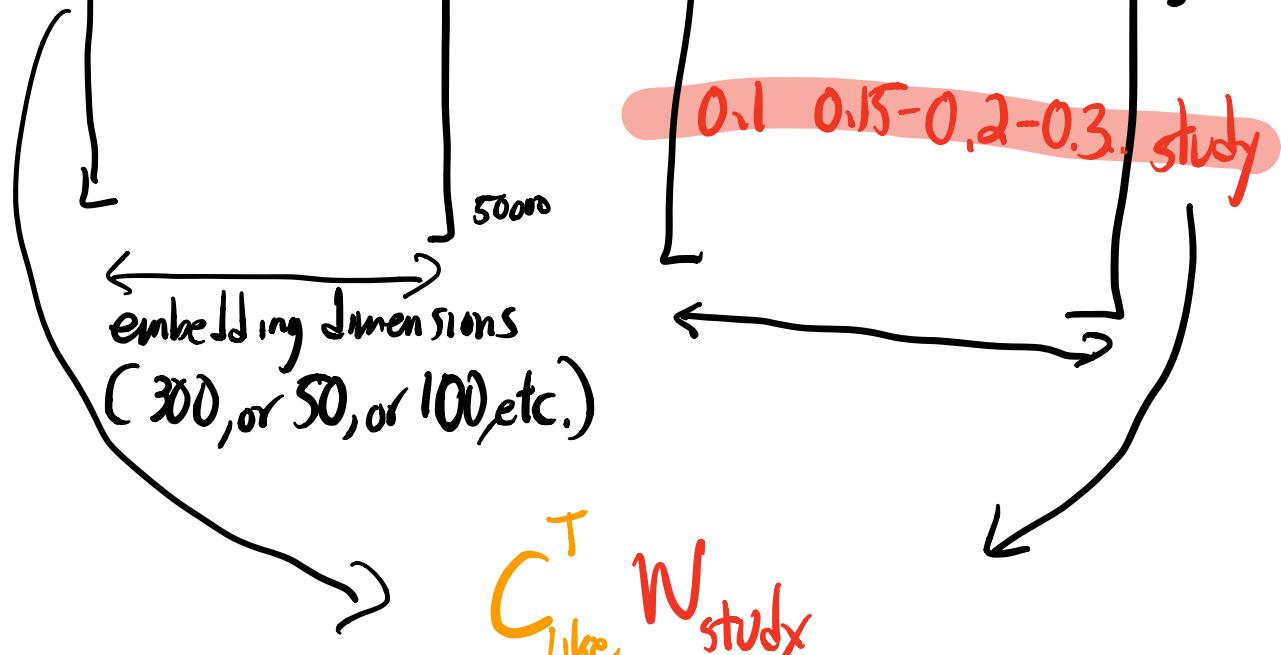
our context matrix (our context vector for "like")

## Context Matrix

0.2	0.1	0.4	...	apple
-1.1	0.1	2.5	...	ant
...				⋮
0.1	0.3	-1.2	...	like

## Word Matrix

1.2	3.4	0.7	apple
1.2	0.1	2.5	ant
⋮	⋮	⋮	⋮



dot product, measures "agreement" between **like** and **study**. If dot product is high, then that **context word** is frequently co-occurring in the same context window as the **target word**. We can

use the dot product as a proxy for measuring the probability of finding the word **like** in the same context window as the word **study**.

However, there's two problems:

- ① dot products can be negative
- ② dot products can be  $\geq 1$

must be between 0 and 1  
and must sum to 1.

So we use the Softmax Function: our original

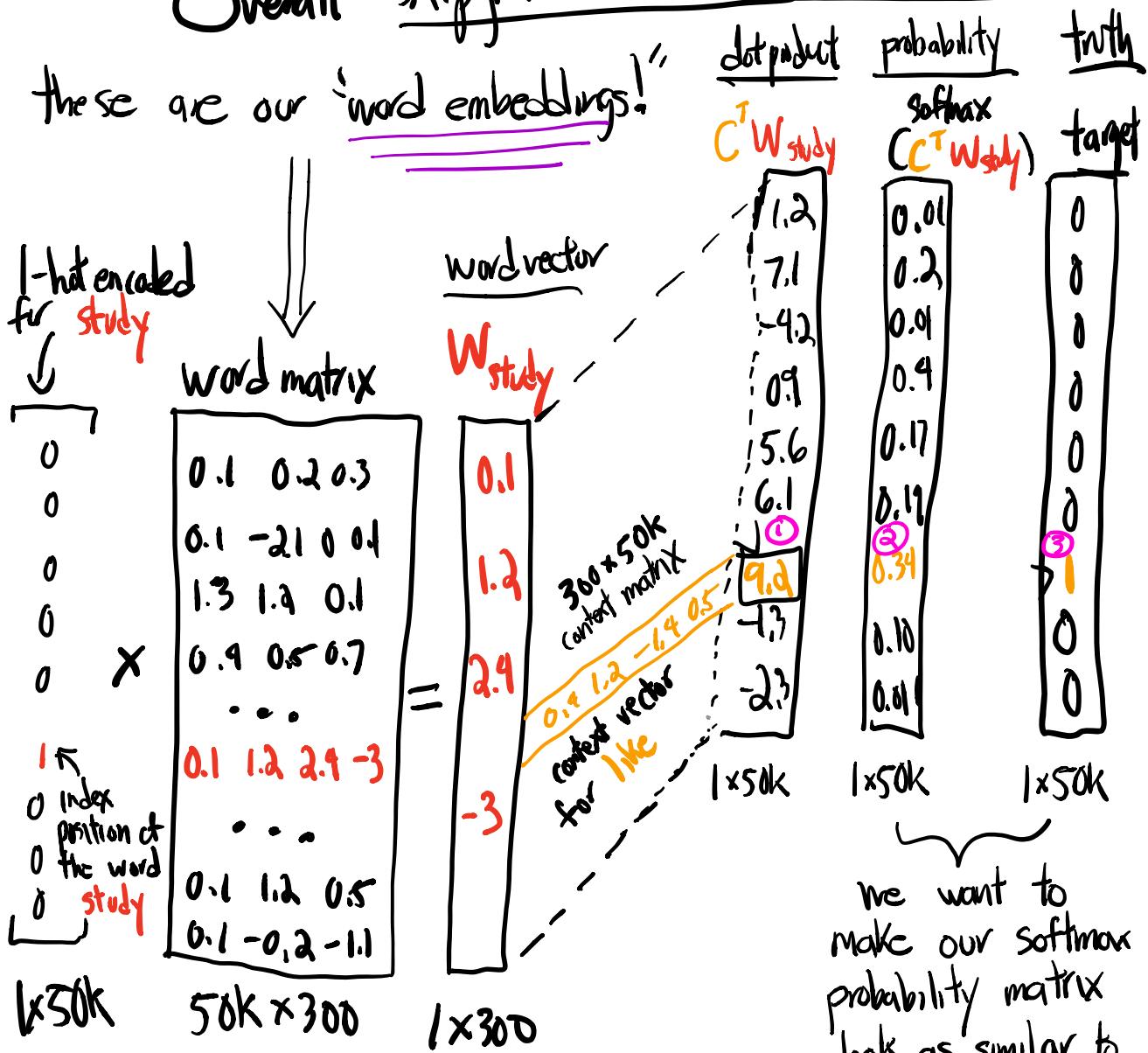
$$p(\text{like} | \text{study}) = \frac{\exp(C_{\text{like}}^T W_{\text{study}})}{\sum_{i=1}^V \exp(C_i^T W_{\text{study}})}$$

we do this  $V$  times  
(50,000 times, for each word in vocab)

level of agreement  
between a random word  
 $i$  and **study**

# Overall skipgram wordvec architecture

these are our 'word embeddings!'



We want to make our softmax probability matrix look as similar to the ground truth matrix as possible

- ① This is the dot product of the word vector for **study** and context vector for **like**.
- ② This is the predicted probability of seeing the word **like** in the context window of **study**.
- ③ This is our target that we are optimizing for

You see in our first pass we had a prediction for  $p(\text{like} | \text{study}) = 0.34$ . The truth was 1. How do we update our parameters? Using derivatives! (more specifically, gradient descent)

$$\text{NLL } J(\theta) = \frac{1}{T} \sum_{t=1}^T \sum_{\substack{-n \leq j \leq m \\ j \neq 0}} \log p(w_{t+j} | w_t, \theta)$$

$$\begin{aligned} & \frac{\partial}{\partial \theta} \log p(w_{t+j} | w_t, \theta) \\ &= \frac{\partial}{\partial \theta} \log \frac{\exp(C_{\tilde{t}}^T w_t)}{\sum_i \exp(C_i^T w_t)} \end{aligned}$$

$\tilde{t}$  = a context word of the word  $w_t$

Important identity:

$$\log \left( \frac{A}{B} \right) = \log(A) - \log(B)$$

Therefore,

$$\textcircled{1} \frac{\partial}{\partial W} \log \frac{\exp(C_{nt}^T W_t)}{\sum_i (\exp(C_i^T W_t))} =$$

$$\textcircled{2} \frac{\partial}{\partial W} \log \underbrace{\exp(C_{nt}^T W_t)}_{\substack{\text{these cancel} \\ \text{each other} \\ \text{out!}}} - \log \sum_i \exp(C_i^T W_t)$$

↓  
↓  
↓

$$\frac{\partial}{\partial W} C_{nt}^T W_t$$

↓  
(using chain rule of  
derivatives and skipping  
several steps)

Important:  
In the lecture,  
I wrote  
 $W_t$  - this  
was NOT  
correct

$$= C_{nt}$$

$$= C_{nt}$$

↑  
the context  
target

$$\begin{aligned} & - \frac{\exp(C_i^T W_t)}{\sum_j \exp(C_j^T W_t)} C_i \\ & - \sum_i p(W_{nt} | W_t) C_i \end{aligned}$$

↓  
weighted average of all  
the context word probabilities

Since we are finding the derivative, and the min of this function comes when

$\frac{\partial J(\theta)}{\partial \theta} = 0$ , we are essentially looking to update  $W$  so that the difference between  $C_{\text{nt}}$  and  $\sum p(W_i | W_t) \cdot C_i$  are equal. This is essentially saying we want to find the best configuration for the word embeddings that best predicts the target word  $W_t$ 's context word  $W_{\text{nt}}$ .