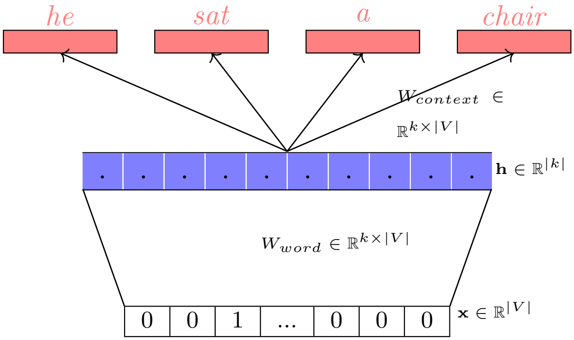
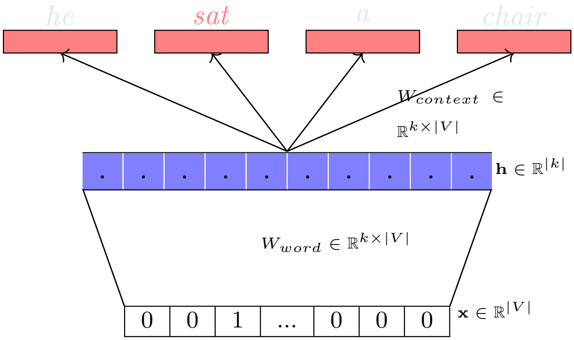
Module 10.5: Skip-gram model

- The model that we just saw is called the continuous bag of words model (it predicts an output word give a bag of context words)

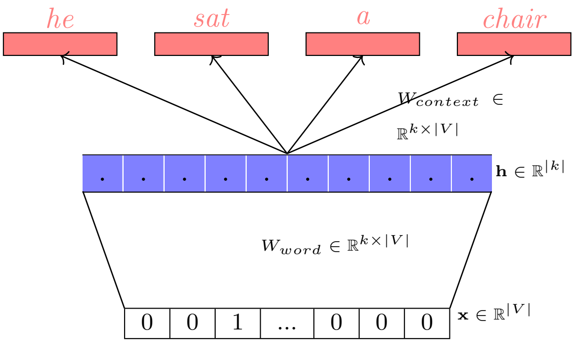
- The model that we just saw is called the continuous bag of words model (it predicts an output word give a bag of context words)
- We will now see the skip gram model (which predicts context words given an input word)



- Notice that the role of *context* and *word* has changed now

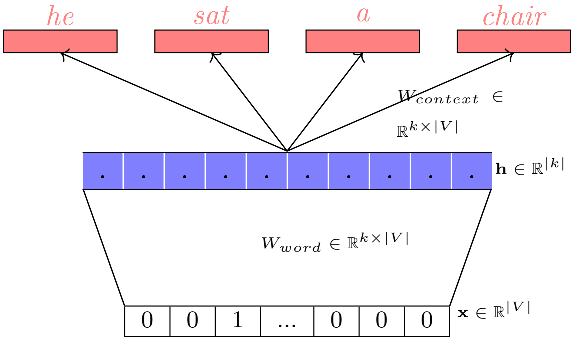


- Notice that the role of *context* and *word* has changed now
- In the simple case when there is only one *context* word, we will arrive at the same update rule for u_c as we did for v_w earlier



- Notice that the role of *context* and *word* has changed now
- In the simple case when there is only one *context* word, we will arrive at the same update rule for u_c as we did for v_w earlier
- Notice that even when we have multiple context words the loss function would just be a summation of many cross entropy errors

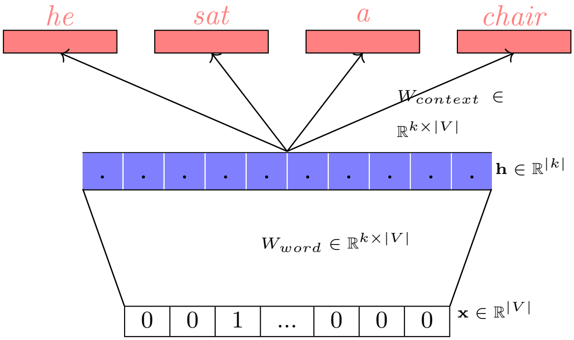
$$\mathscr{L}(\theta) = -\sum_{i=1}^{d-1} \log \hat{y}_{w_i}$$



- Notice that the role of *context* and *word* has changed now
- In the simple case when there is only one *context* word, we will arrive at the same update rule for u_c as we did for v_w earlier
- Notice that even when we have multiple context words the loss function would just be a summation of many cross entropy errors

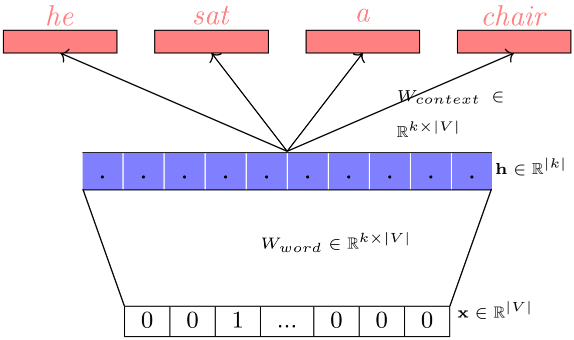
$$\mathscr{L}(\theta) = -\sum_{i=1}^{d-1} \log \hat{y}_{w_i}$$

- Typically, we predict context words on both sides of the given word



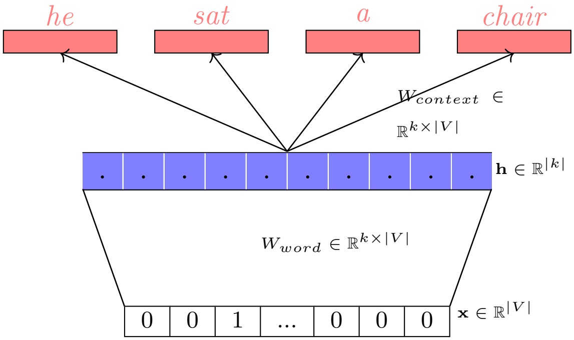
Some problems

- Same as bag of words



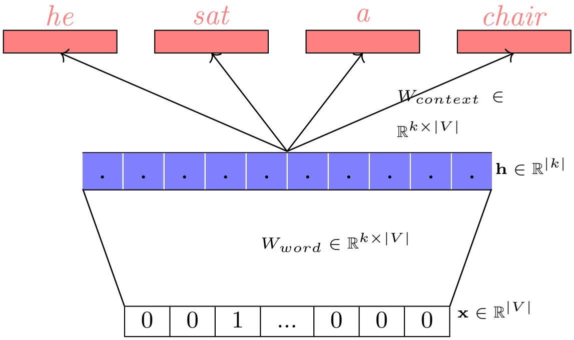
Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive



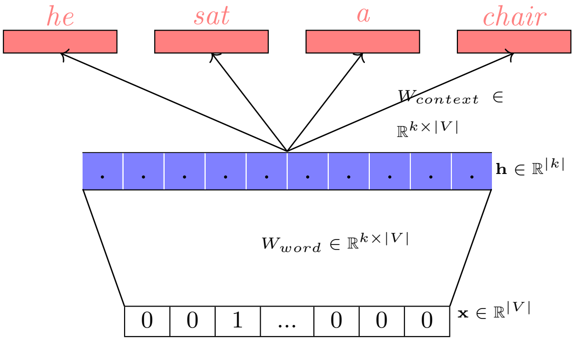
Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive
- Solution 1: Use negative sampling



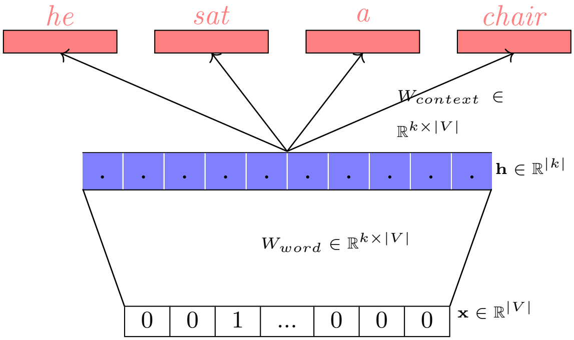
Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive
- Solution 1: Use negative sampling
- Solution 2: Use contrastive estimation



Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive
- Solution 1: Use negative sampling
- Solution 2: Use contrastive estimation
- Solution 3: Use hierarchical softmax



Some problems

- Same as bag of words
- The softmax function at the output is computationally expensive
- Solution 1: Use negative sampling
- Solution 2: Use contrastive estimation
- Solution 3: Use hierarchical softmax

- D = [(sat, on), (sat, a), (sat, chair), (on, a), (on,chair), (a,chair), (on,sat), (a, sat), (chair,sat), (a, on), (chair, on), (chair, a)]

- Let D be the set of all correct (w, c) pairs in the corpus

- D = [(sat, on), (sat, a), (sat, chair), (on, a), (on,chair), (a,chair), (on,sat), (a, sat), (chair,sat), (a, on), (chair, on), (chair, a)]
- D' = [(sat, oxygen), (sat, magic), (chair, sad), (chair, walking)]

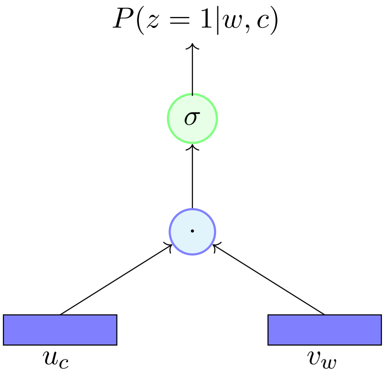
- Let D be the set of all correct (w, c) pairs in the corpus
- Let D' be the set of all incorrect (w, r) pairs in the corpus

- D = [(sat, on), (sat, a), (sat, chair), (on, a), (on,chair), (a,chair), (on,sat), (a, sat), (chair,sat), (a, on), (chair, on), (chair, a)]
- D' = [(sat, oxygen), (sat, magic), (chair, sad), (chair, walking)]

- Let D be the set of all correct (w, c) pairs in the corpus
- Let D' be the set of all incorrect (w, r) pairs in the corpus
- D' can be constructed by randomly sampling a context word r which has never appeared with w and creating a pair (w, r)

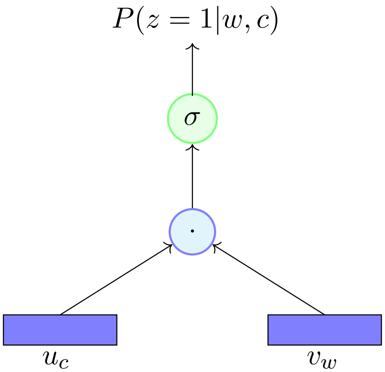
- D = [(sat, on), (sat, a), (sat, chair), (on, a), (on,chair), (a,chair), (on,sat), (a, sat), (chair,sat), (a, on), (chair, on), (chair, a)]
- D' = [(sat, oxygen), (sat, magic), (chair, sad), (chair, walking)]

- Let D be the set of all correct (w, c) pairs in the corpus
- Let D' be the set of all incorrect (w, r) pairs in the corpus
- D' can be constructed by randomly sampling a context word r which has never appeared with w and creating a pair (w, r)
- As before let v_w be the representation of the word w and u_c be the representation of the context word c



- For a given $(w, c) \in D$ we are interested in maximizing

$$p(z = 1|w, c)$$

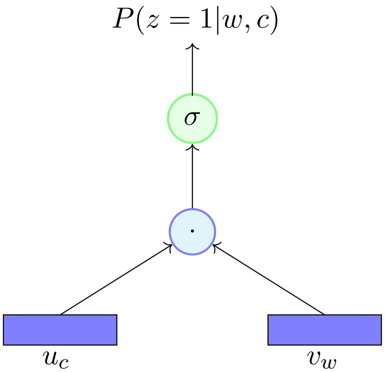


- For a given $(w, c) \in D$ we are interested in maximizing

$$p(z = 1|w, c)$$

- Let us model this probability by

$$\begin{aligned} p(z = 1|w, c) &= \sigma(u_c^T v_w) \\ &= \frac{1}{1 + e^{-u_c^T v_w}} \end{aligned}$$



- For a given $(w,c) \in D$ we are interested in maximizing

$$p(z=1|w,c)$$

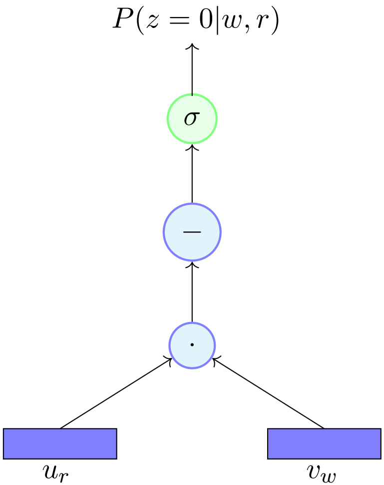
- Let us model this probability by

$$p(z=1|w,c) = \sigma(u_c^T v_w) = \frac{1}{1+e^{-u_c^T v_w}}$$

- Considering all $(w,c) \in D$, we are interested in

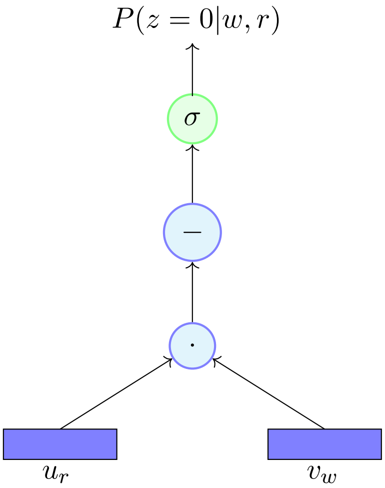
$$\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z=1|w,c)$$

where θ is the word representation (v_w) and context representation (u_c) for all words in our corpus



- For $(w,r) \in D'$ we are interested in maximizing

$$p(z=0|w,r)$$

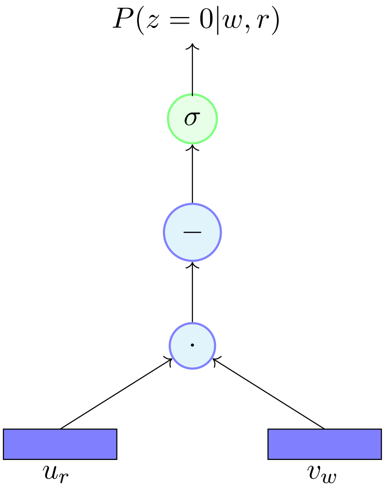


- For $(w,r) \in D'$ we are interested in maximizing

$$p(z=0|w,r)$$

- Again we model this as

$$p(z=0|w,r) = 1 - \sigma(u_r^T v_w)$$

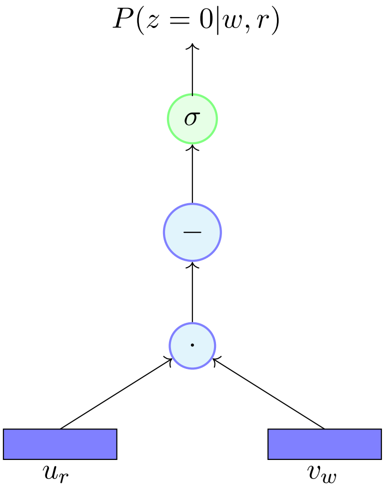


- For $(w, r) \in D'$ we are interested in maximizing

$$p(z = 0|w, r)$$

- Again we model this as

$$p(z = 0|w, r) = 1 - \sigma(u_r^T v_w)$$
$$= 1 - \frac{1}{1 + e^{-v_r^T v_w}}$$

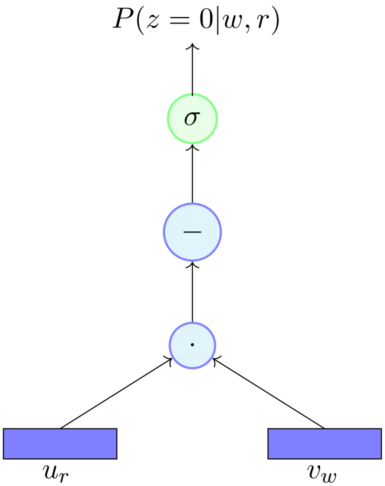


- For $(w, r) \in D'$ we are interested in maximizing

$$p(z = 0|w, r)$$

- Again we model this as

$$\begin{aligned} p(z = 0|w, r) &= 1 - \sigma(u_r^T v_w) \\ &= 1 - \frac{1}{1 + e^{-v_r^T v_w}} \\ &= \frac{1}{1 + e^{u_r^T v_w}} \end{aligned}$$

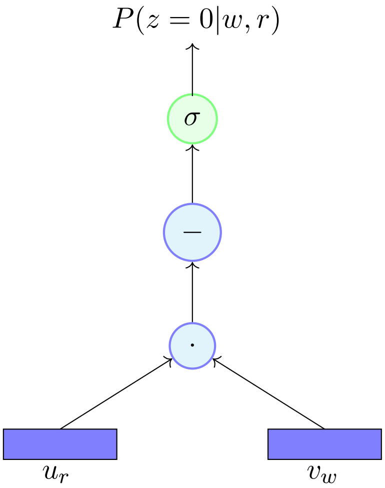


- For $(w, r) \in D'$ we are interested in maximizing

$$p(z = 0|w, r)$$

- Again we model this as

$$
\begin{aligned}
p(z = 0|w, r) &= 1 - \sigma(u_r^T v_w) \\
&= 1 - \frac{1}{1 + e^{-v_r^T v_w}} \\
&= \frac{1}{1 + e^{u_r^T v_w}} = \sigma(-u_r^T v_w)
\end{aligned}
$$



- For $(w, r) \in D'$ we are interested in maximizing

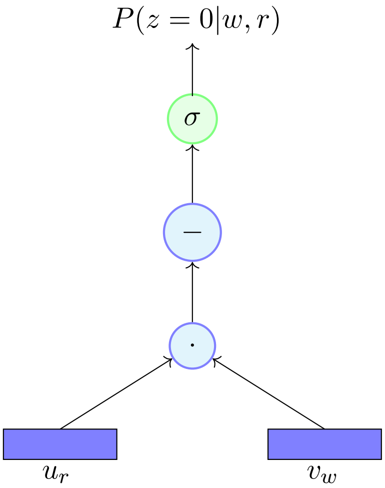
$$p(z = 0|w, r)$$

- Again we model this as

$$\begin{aligned} p(z = 0|w, r) &= 1 - \sigma(u_r^T v_w) \\ &= 1 - \frac{1}{1 + e^{-v_r^T v_w}} \\ &= \frac{1}{1 + e^{u_r^T v_w}} = \sigma(-u_r^T v_w) \end{aligned}$$

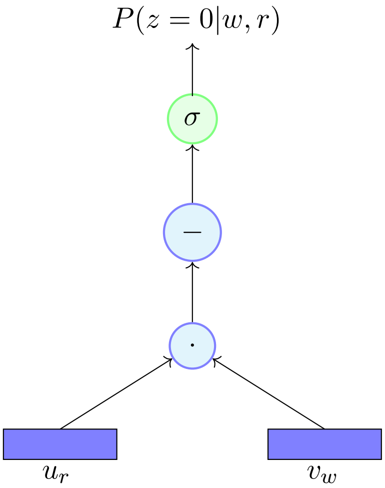
- Considering all $(w, r) \in D'$, we are interested in

$$\underset{\theta}{maximize} \prod_{(w,r) \in D'} p(z = 0|w, r)$$



- Combining the two we get:

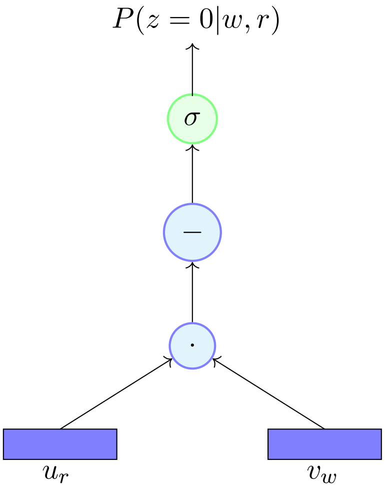
$$\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z = 1|w, c) \prod_{(w,r)\in D'} p(z = 0|w, r)$$



- Combining the two we get:

$$\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z = 1|w, c) \prod_{(w,r)\in D'} p(z = 0|w, r)$$

$$= \underset{\theta}{maximize} \prod_{(w,c)\in D} p(z = 1|w, c) \prod_{(w,r)\in D'} (1 - p(z = 1|w, r))$$

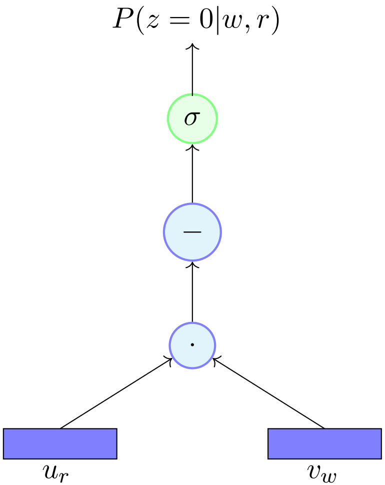


- Combining the two we get:

$$\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z=1|w,c) \prod_{(w,r)\in D'} p(z=0|w,r)$$

$$= \underset{\theta}{maximize} \prod_{(w,c)\in D} p(z=1|w,c) \prod_{(w,r)\in D'} (1 - p(z=1|w,r))$$

$$= \underset{\theta}{maximize} \sum_{(w,c)\in D} \log p(z=1|w,c) + \sum_{(w,r)\in D'} \log(1 - p(z=1|w,r))$$



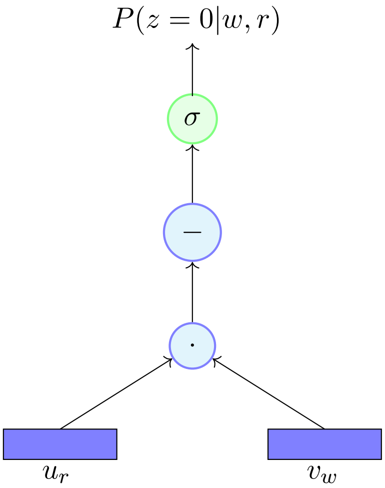
- Combining the two we get:

$$\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z = 1|w, c) \prod_{(w,r)\in D'} p(z = 0|w, r)$$

$$= \underset{\theta}{maximize} \prod_{(w,c)\in D} p(z = 1|w, c) \prod_{(w,r)\in D'} (1 - p(z = 1|w, r))$$

$$= \underset{\theta}{maximize} \sum_{(w,c)\in D} \log p(z = 1|w, c) + \sum_{(w,r)\in D'} \log(1 - p(z = 1|w, r))$$

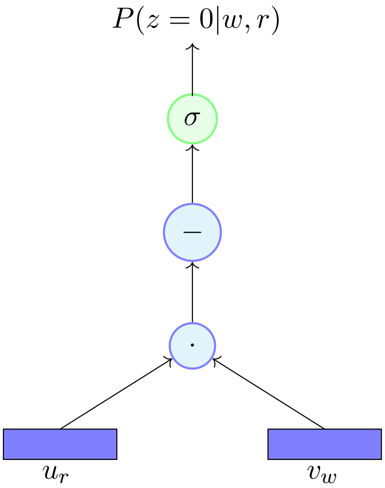
$$= \underset{\theta}{maximize} \sum_{(w,c)\in D} \log \frac{1}{1 + e^{-v_c^T v_w}} + \sum_{(w,r)\in D'} \log \frac{1}{1 + e^{v_r^T v_w}}$$



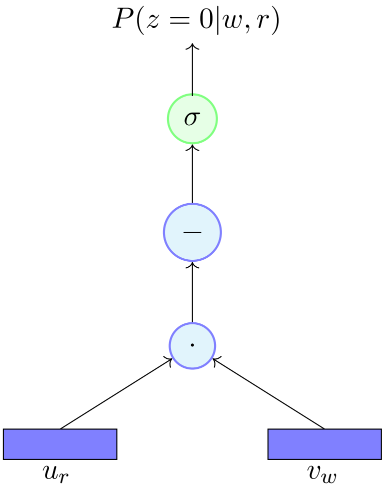
- Combining the two we get:

$$
\begin{aligned}
&\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z=1|w,c) \prod_{(w,r)\in D'} p(z=0|w,r)\\
=&\underset{\theta}{maximize} \prod_{(w,c)\in D} p(z=1|w,c) \prod_{(w,r)\in D'} (1-p(z=1|w,r))\\
=&\underset{\theta}{maximize} \sum_{(w,c)\in D} \log p(z=1|w,c)\\
&\quad + \sum_{(w,r)\in D'} \log(1-p(z=1|w,r))\\
=&\underset{\theta}{maximize} \sum_{(w,c)\in D} \log \frac{1}{1+e^{-v_c^T v_w}} + \sum_{(w,r)\in D'} \log \frac{1}{1+e^{v_r^T v_w}}\\
=&\underset{\theta}{maximize} \sum_{(w,c)\in D} \log \sigma(v_c^T v_w) + \sum_{(w,r)\in D'} \log \sigma(-v_r^T v_w)
\end{aligned}
$$

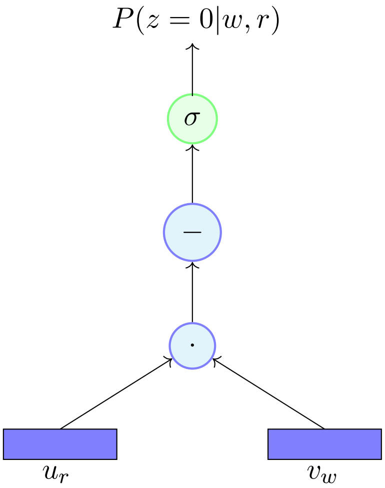
where $\sigma(x) = \frac{1}{1+e^{-x}}$



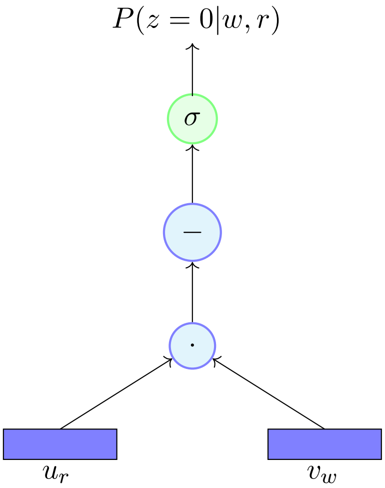
- In the original paper, *Mikolov et. al.* sample k negative (w, r) pairs for every positive (w, c) pairs



- In the original paper, *Mikolov et. al.* sample k negative (w, r) pairs for every positive (w, c) pairs
- The size of D' is thus k times the size of D



- In the original paper, *Mikolov et. al.* sample k negative (w, r) pairs for every positive (w, c) pairs
- The size of D' is thus k times the size of D
- The random context word is drawn from a modified unigram distribution



- In the original paper, *Mikolov et. al.* sample k negative (w, r) pairs for every positive (w, c) pairs
- The size of D' is thus k times the size of D
- The random context word is drawn from a modified unigram distribution

$$r \sim p(r)^{\frac{3}{4}}$$

$$r \sim \frac{count(r)^{\frac{3}{4}}}{N}$$

N = total number of words in the corpus