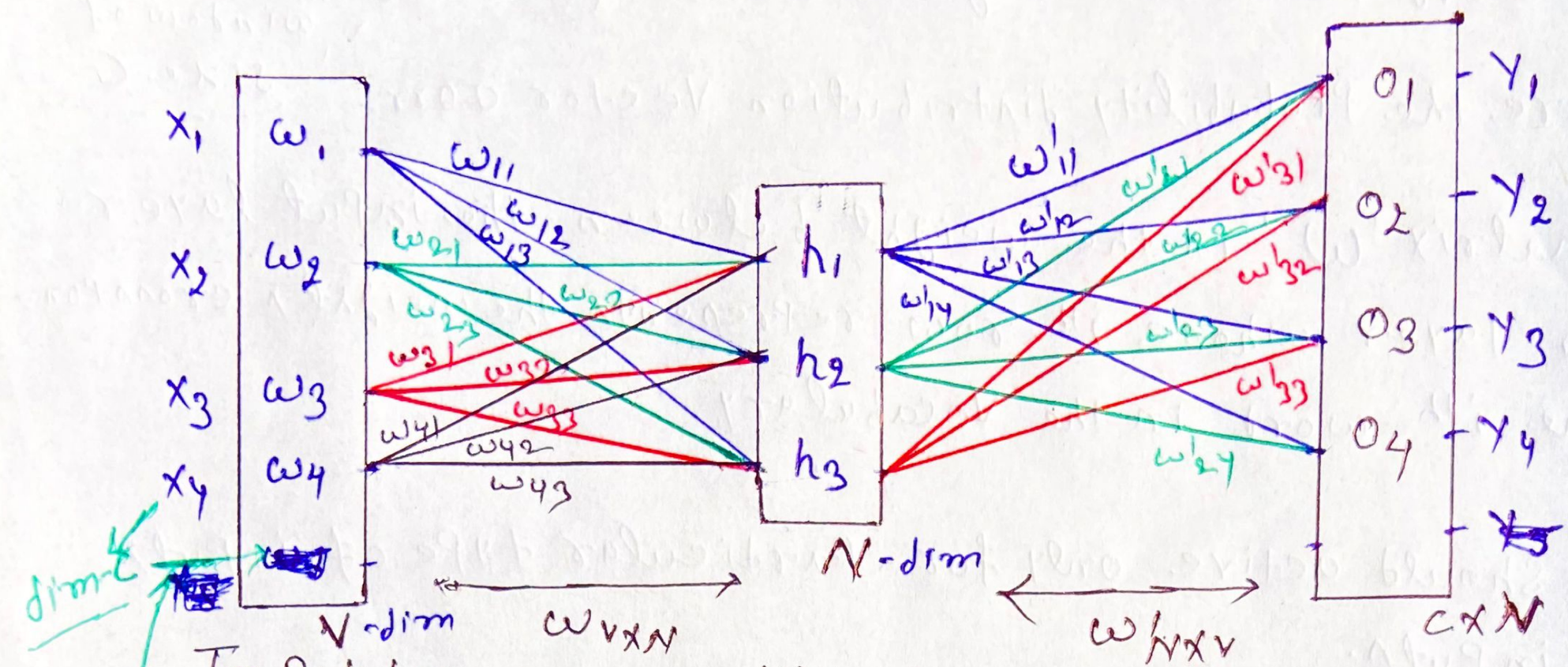
Continuous bag of words (CBOW)



(Simple CBOW model with one word (C=1) in the Context)

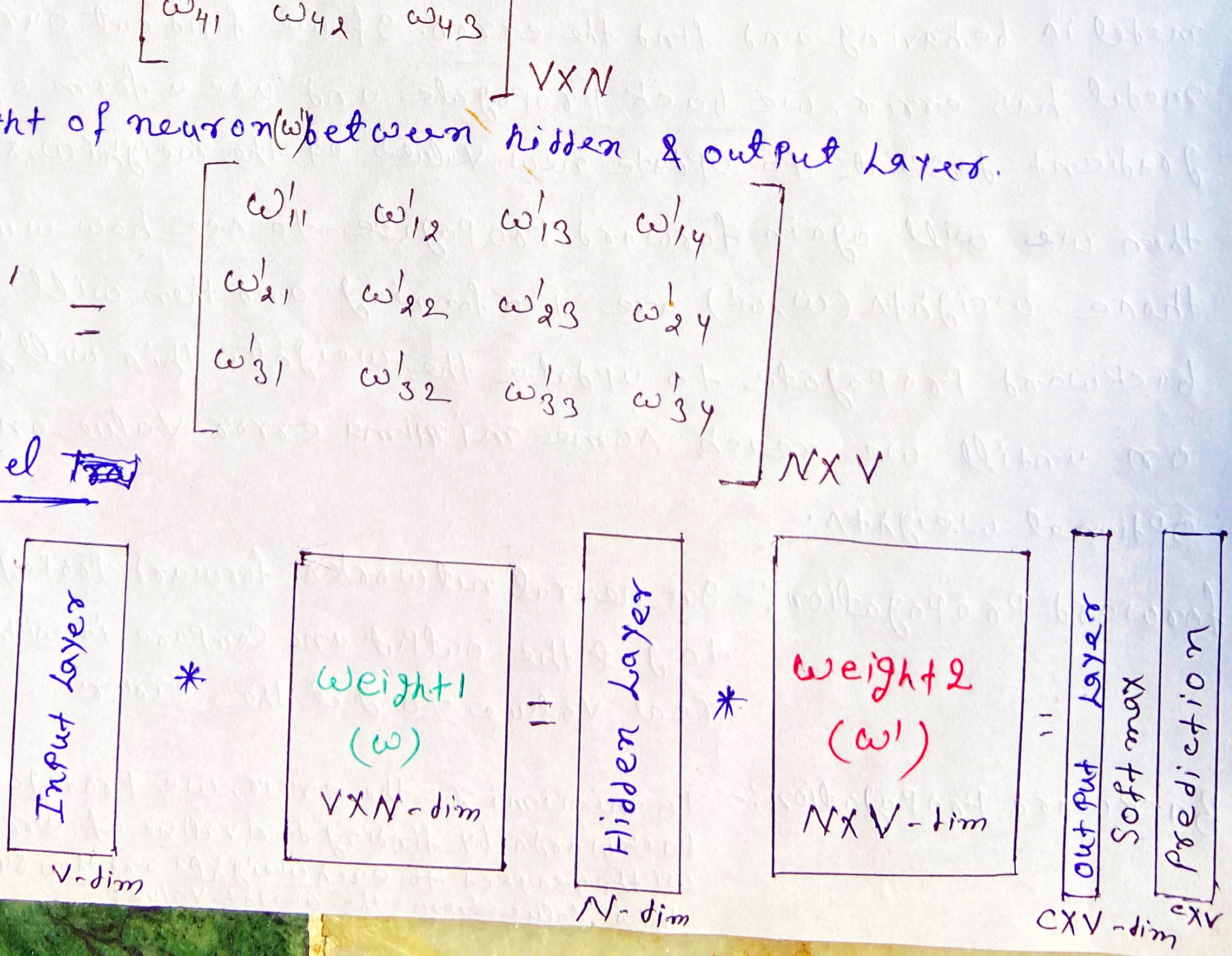
weight of neuron(w) between Input Layer & hidden layer

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{bmatrix}_{VXN}$$

weight of neuron(w') between hidden & output Layer.

$$W' = \begin{bmatrix} w'_{11} & w'_{12} & w'_{13} & w'_{14} \\ w'_{21} & w'_{22} & w'_{23} & w'_{24} \\ w'_{31} & w'_{32} & w'_{33} & w'_{34} \end{bmatrix}_{NXV}$$

Model

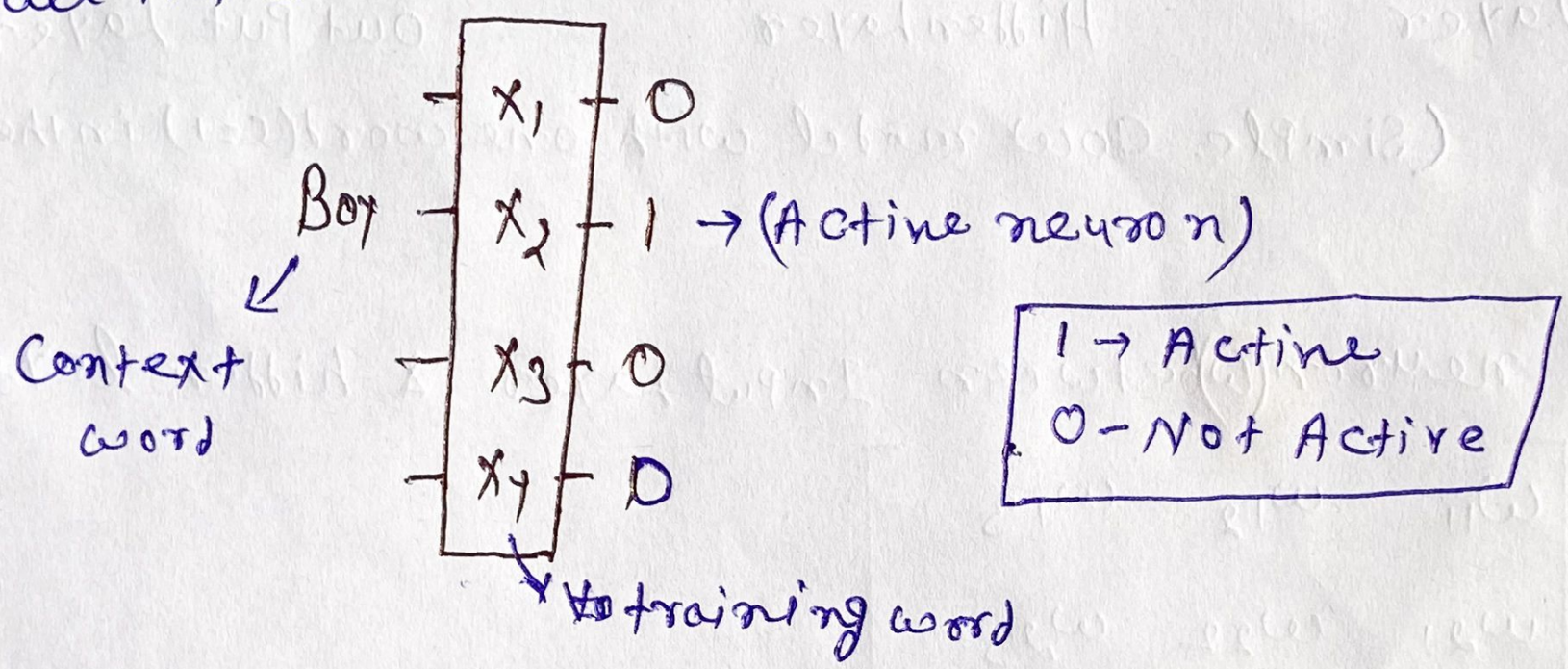


→ In this model 'x' represent the one-hot encoded vector corresponding to the input word in the training instance. $\{X_1, \ldots X_c\}$ for a word window of size C

→ $\{Y_1 \ldots Y_c\}$ are the probability distribution vector

→ The (V x N) matrix 'W' is the weight between the input layer and hidden layer whose ith row represents the weights corresponding to the ith word in the vocabulary.

Each neuron should active only for particular type of inputs and not all inputs.



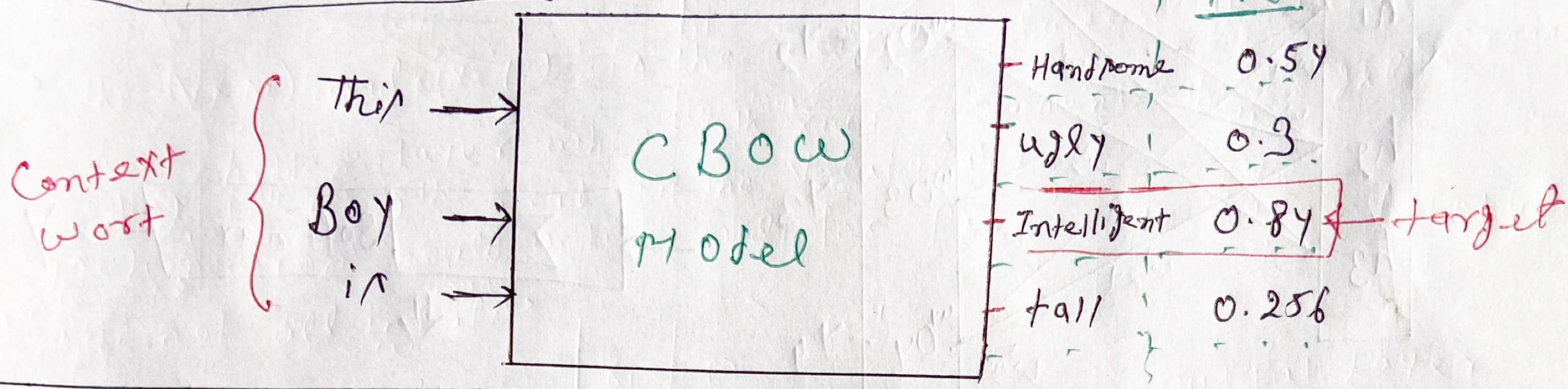
Therefore, by Propagation forward we see how how well our model is behaving and find the error. If we find out our model has error, we back Propagate and use a form of gradient descent to update new values of the weights (w, & w'). then we will again forward Propagate to see how well those weights (w, w') are performing and then will backward Propagate to update the weights. This will go on untill we reach some minima error value and optimal weights.

forward Propogation :- In neural networks, forward Propagate to get the output and compare it with the real value to get the error

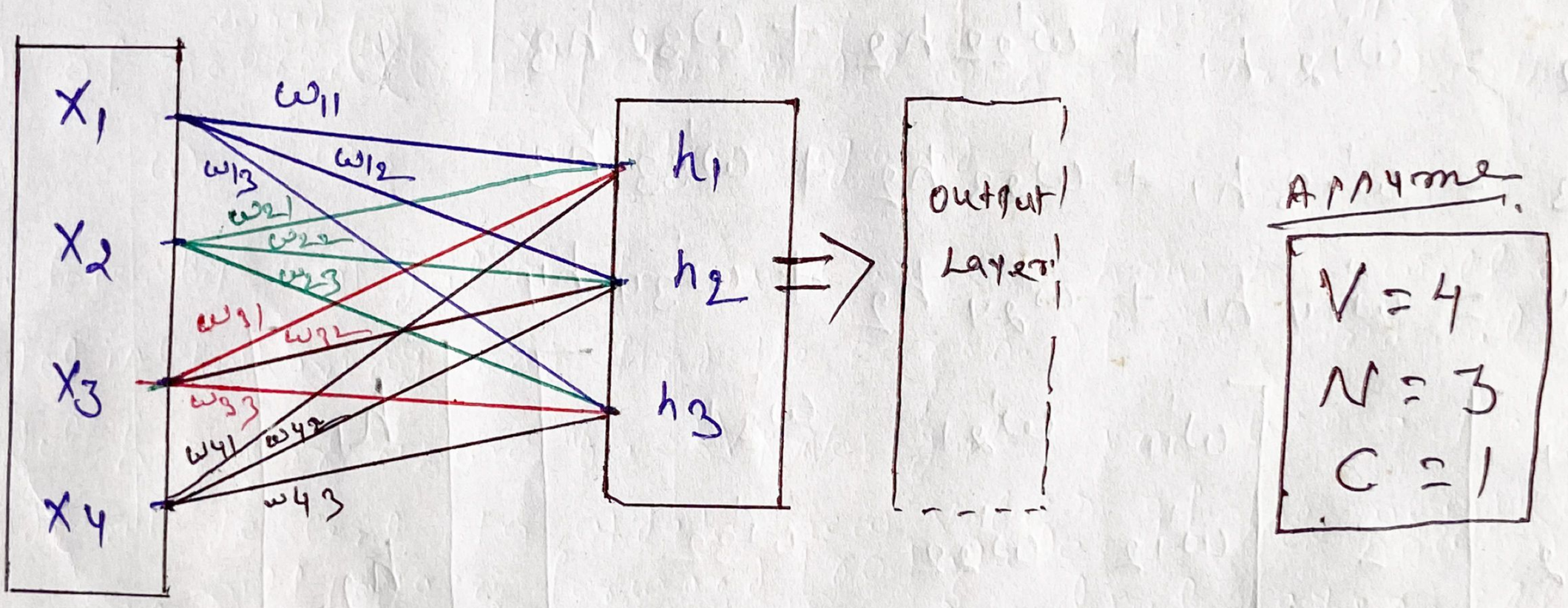
Backward Propagation :- To minimize the error, we Propagate backwards by finding derivative of error with respect to each weights and then subtracting this value from the weight value.

After getting optimal weight of neuron to we predict the target word. output will provide the probability distribution scores corresponding to the word.

"This boy is ____."



Forward Propagation : Weight calculation (w) :- Input Layer to hidden Layer



$$h_1 = w_{11}x_1 + w_{21}x_2 + w_{31}x_3 + w_{41}x_4$$

$$h_2 = w_{12}x_2 + w_{22}x_2 + w_{32}x_3 + w_{42}x_4$$

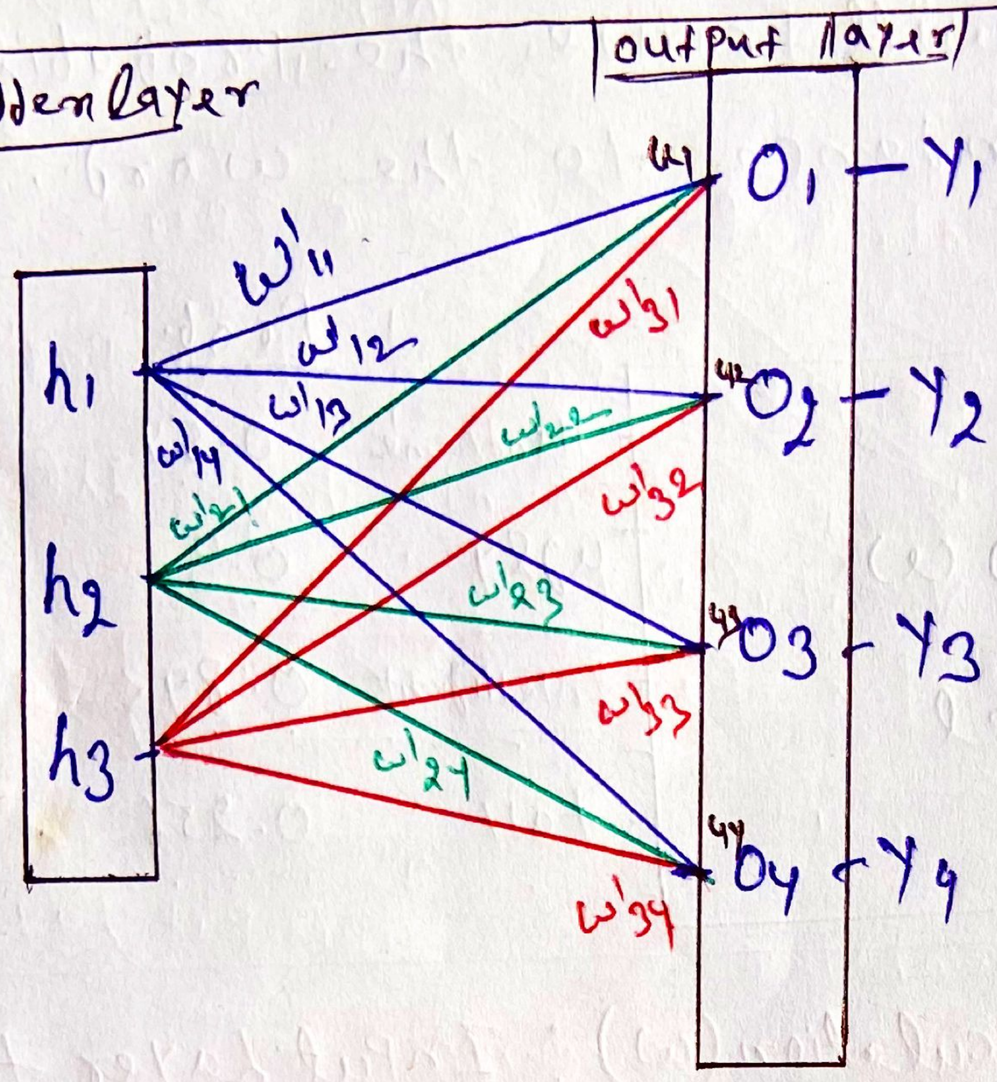
$$h_3 = w_{13}x_3 + w_{23}x_2 + w_{33}x_3 + w_{43}x_4$$

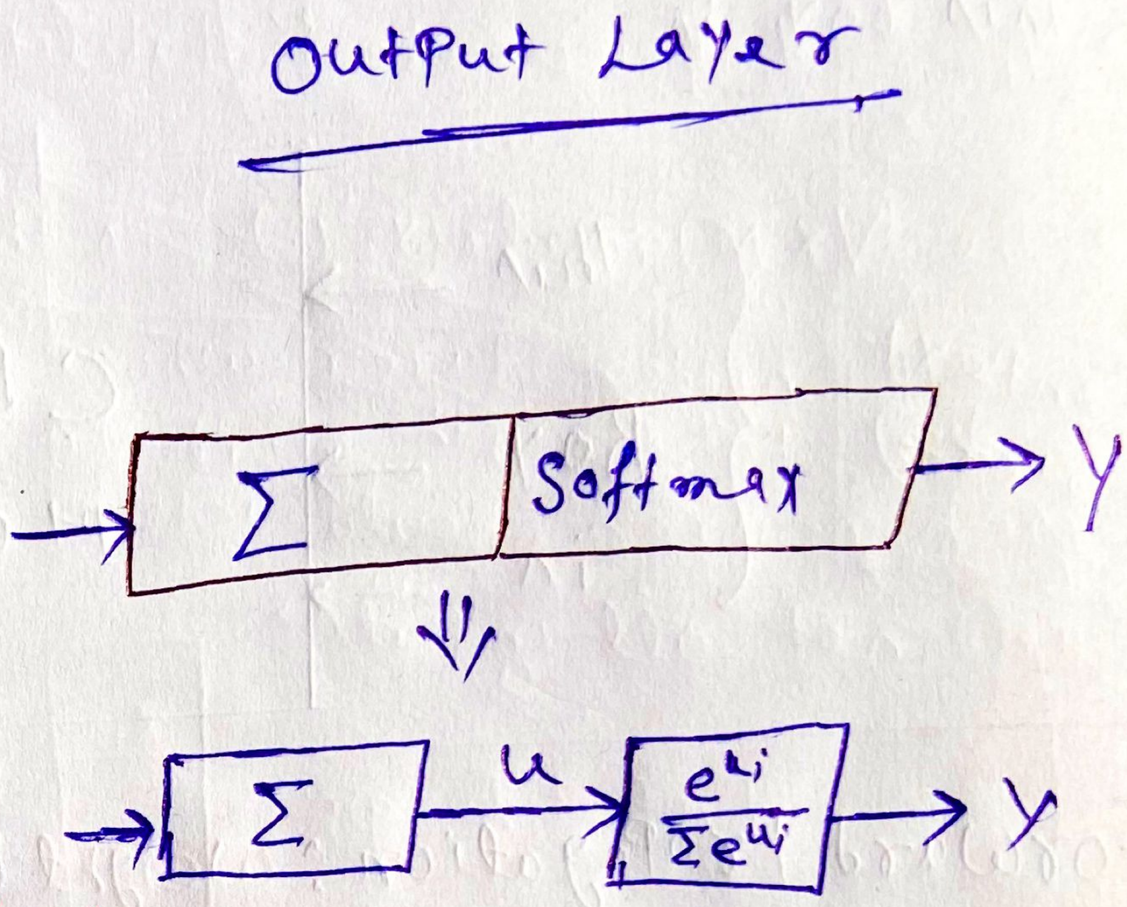
$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \\ w_{13} & w_{23} & w_{33} & w_{43} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$N = 3$; $N \times N$; ~~3x3~~ 3×4 ; $V \times C$; ~~4~~ 4×1

$$h = w^T x \qquad (1)$$

Forward Calculation: Weight Calculation (w') :- Hidden to O/P Layer.





$$u_1 = w'_{11} h_1 + w'_{21} h_2 + w'_{31} h_3$$

$$u_2 = w'_{12} h_1 + w'_{22} h_2 + w'_{32} h_3$$

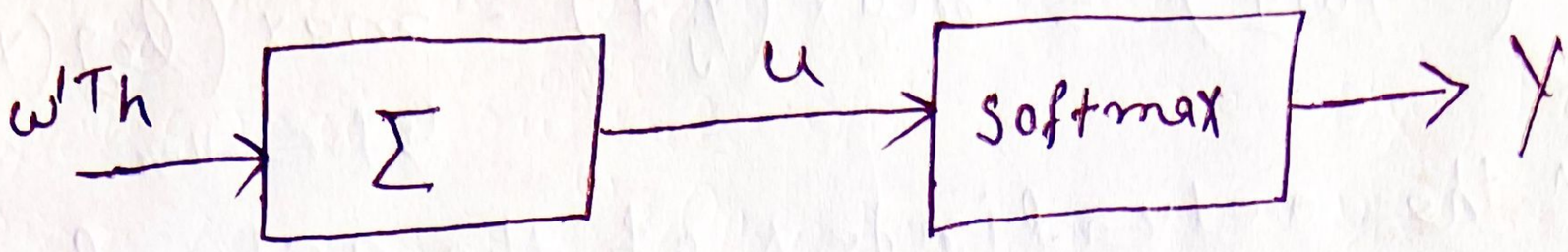
$$u_3 = w'_{13} h_1 + w'_{23} h_2 + w'_{33} h_3$$

$$u_4 = w'_{14} h_1 + w'_{24} h_2 + w'_{34} h_4$$

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} w'_{11} & w'_{21} & w'_{31} \\ w'_{12} & w'_{22} & w'_{32} \\ w'_{13} & w'_{23} & w'_{33} \\ w'_{14} & w'_{24} & w'_{34} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

$$\boxed{u = w'^T h} \longrightarrow (2)$$

output layer



Softmax function:-

$$f(x_i) = \frac{e^{x_j}}{\sum_{i=1}^{V} e^{x_i}}$$

$$y_1 = \frac{e^{u_1}}{e^{u_1} + e^{u_2} + e^{u_3} + e^{u_4}}$$

$$y_2 = \frac{e^{u_2}}{e^{u_1} + e^{u_2} + e^{u_3} + e^{u_4}}$$

The Softmax output for the word w_j with respect to the given input context word w_I can be given by Conditional Probability, in terms of exponentials, i.e.

$$P(w_j/w_I) = y_j = \frac{e^{u_j}}{\sum_{j=1}^{V} e^{u_j}} \quad \text{— (3)}$$

Error Calculation

Assumption:- Let 'w_o' is the actual output word, w_I is the given input context words and 'V' is the size of the input context.

The objective is to maximize the Conditional Probability of observing the actual output word 'w_o', given the input context words 'w_I', thus the Loss function can be defined as:-

$$\max P(\omega_O/\omega_I) = \max(y_{j*}) = \max(\log(y_{j*}))$$

where, $j*$ is the ~~actual~~ index of the actual output word in the output layer.

Example:- If 4th word represent the o/p

$$E(Y_4) = \log P(\omega_{Y4}/\omega_I) = \log_e\left(\frac{e^{u_4}}{e^{u_1}+e^{u_2}+e^{u_3}+e^{u_4}}\right)$$

$$E(Y_4) = u_4 - \log(e^{u_1}+e^{u_2}+e^{u_3}+e^{u_4})$$

As, we want to minimize the error, $\Rightarrow 0$.

$$E = -\log P(\omega_O/\omega_I)$$

$$E(Y_4) = \log(e^{u_1}+e^{u_2}+e^{u_3}+e^{u_4}) - u_4$$

we can generalize it as:-

$$\boxed{E = \log\sum_{j=1}^{V} e^{u_j} - u_{j*}} \quad \text{--- (4)}$$

It is the special case of cross-entropy measurement b/w two probabilistic distribution u_j and u_{j*}

taking derivative E w.r.to u_4, we have

$$\frac{dE(Y_4)}{d(u_4)} = \frac{e^{u_4}}{e^{u_1}+e^{u_2}+e^{u_3}+e^{u_4}} - \frac{d(u_4)}{d(u_4)}$$

$$\boxed{\frac{d\log x}{dx} = \frac{1}{x} \qquad \frac{de^x}{dx} = e^x}$$

$$\frac{dE(Y_4)}{d(u_4)} = Y_4 - 1$$

we can generalize the above derivation.

$$\boxed{\frac{dE}{du_j} = \frac{d(\log\sum_{i=1}^{V} e^{u_i} - u_{i*})}{du_j} = Y_j - t_j = e_j} \quad \text{--- (5)}$$

Note:- $t_j = 1$, if $(t_j = t_{j*})$; else $t_j = 0$

Back-Propagation: Output to hidden layer (Update ω')

In this back-propagation we, update the weight of all neuron from hidden layer to output layer.

Step 1: Taking gradient 'E' w.r.t; ω'_{11}

$$\frac{dE(Y_1)}{d\omega'_{11}} = \frac{dE(Y_1)}{du_1} \times \frac{du_1}{d\omega'_{11}}$$

$$\frac{dE(Y_1)}{du_1} = (Y_1 - O) = e_1 \Rightarrow \text{from eqn (5)}$$

from eqn (2) ↓

$$\frac{du_1}{d\omega'_{11}} = \frac{d(\omega'_{11}h_1 + \omega'_{21}h_2 + \omega'_{31}h_3)}{d\omega'_{11}} = h_1 \quad \left[u_1 = \omega'_{11}h_1 + \omega'_{21}h_2 + \omega'_{31}h_3\right]$$

$$\therefore \frac{dE(Y_1)}{d\omega'_{11}} = \frac{dE(Y_1)}{du_1} \times \frac{du_1}{d\omega'_{11}}$$

$$\boxed{\frac{dE(Y_1)}{d\omega'_{11}} = e_1 \cdot h_1} \quad \longrightarrow (6)$$

Step 2: Updating the weight of ω'_{11}

$$(\omega'_{11})^{new} = \omega'_{11} - \eta \frac{dE(Y_1)}{d\omega'_{11}} = \omega'_{11} - \eta(e_1 \cdot h_1)$$

Similarly, we can update the weight of $\omega'_{12}, \omega'_{13}, \ldots \omega'_{34}$

$$\boxed{(\omega'_{ij})^{new} = (\omega_{ij})^{old} - \eta e_j \cdot h_i} \quad \longrightarrow (7)$$

where, η = learning rate

$e_j = Y_j - t_j$

Back-Propagation: Hidden to Input layer (update w)

In this back propagation we, update the weight of all neurons from hidden layer to output layer.

Step 1: Taking gradient 'E' w.r.t, w_{11}

$$\frac{dE}{dw_{11}} = \frac{dE}{dh_1} \times \frac{dh_1}{dw_{11}}$$

$$\frac{dE}{dh_1} = \left(\frac{dE}{du_1} \times \frac{du_1}{dh_1}\right) + \left(\frac{dE}{du_2} \times \frac{du_2}{dh_2}\right) + \left(\frac{dE}{du_3} \times \frac{du_3}{dh_3}\right) + \left(\frac{dE}{du_4} \times \frac{du_4}{dh_4}\right)$$

$$\frac{dE}{dh_1} = (e_1 \cdot w'_{11}) + (e_2 w'_{12}) + (e_3 w'_{13}) + (e_4 w'_{14}) \quad \text{---(6)}$$

$$\frac{dE}{dh_1} = \sum_{i=1}^{4} e_j w'_{ij} = EH_i \quad \text{---(9)}$$

$$\frac{dh_1}{dw_{11}} = \frac{d(w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4)}{dw_{11}} = x_1 \quad \text{---(10)}$$

$$\frac{\partial E}{\partial w_{11}} = \frac{dE}{dh_1} \times \frac{dh_1}{dw_{11}}$$

$$= \sum_{i=1}^{4} e_j w'_{11} \cdot x_1 = EH_1 \cdot x_1$$

$$\therefore \frac{\partial E}{\partial w_{kj}} = EH_i \cdot x_k \quad \text{---(7)}$$

Step 2: Updating the weight of w_{11}

$$(w_{11})^{new} = (w_{11})^{old} - \eta \frac{dE}{dw_{11}} = (w_{11})^{old} - \eta EH_1 x_1$$

$$(w_{ij})^{new} = (w_{ij})^{old} - \eta EH_i \cdot x_i \quad \text{---(8)}$$

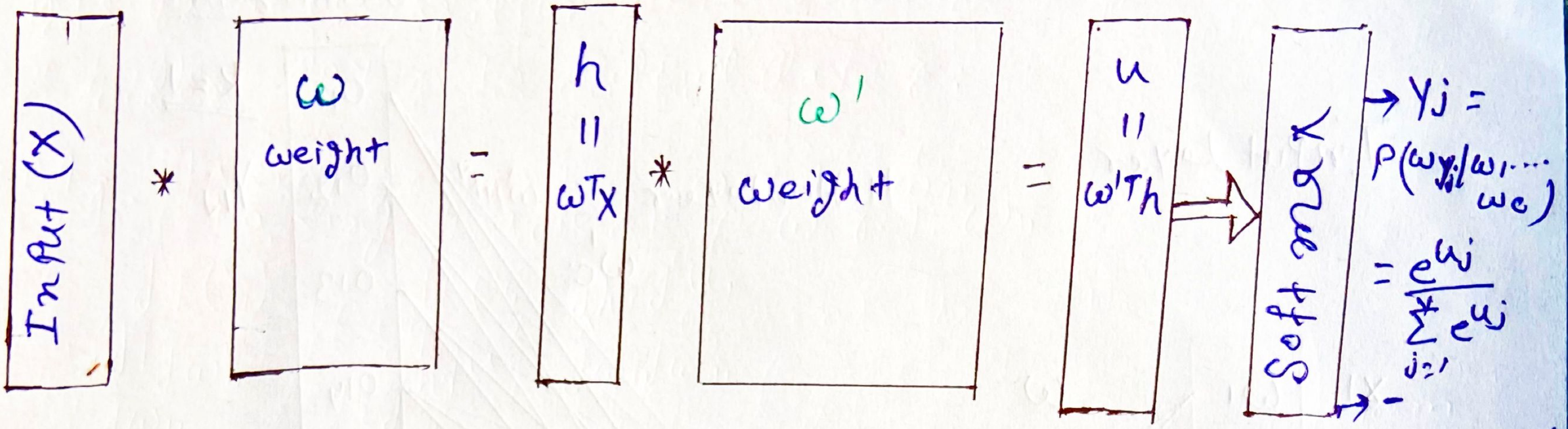
where,

η = learning rate

$EH_i = \sum_{i=1}^{4} e_j w'_{ij}$

CBOW Model

forward propagation →

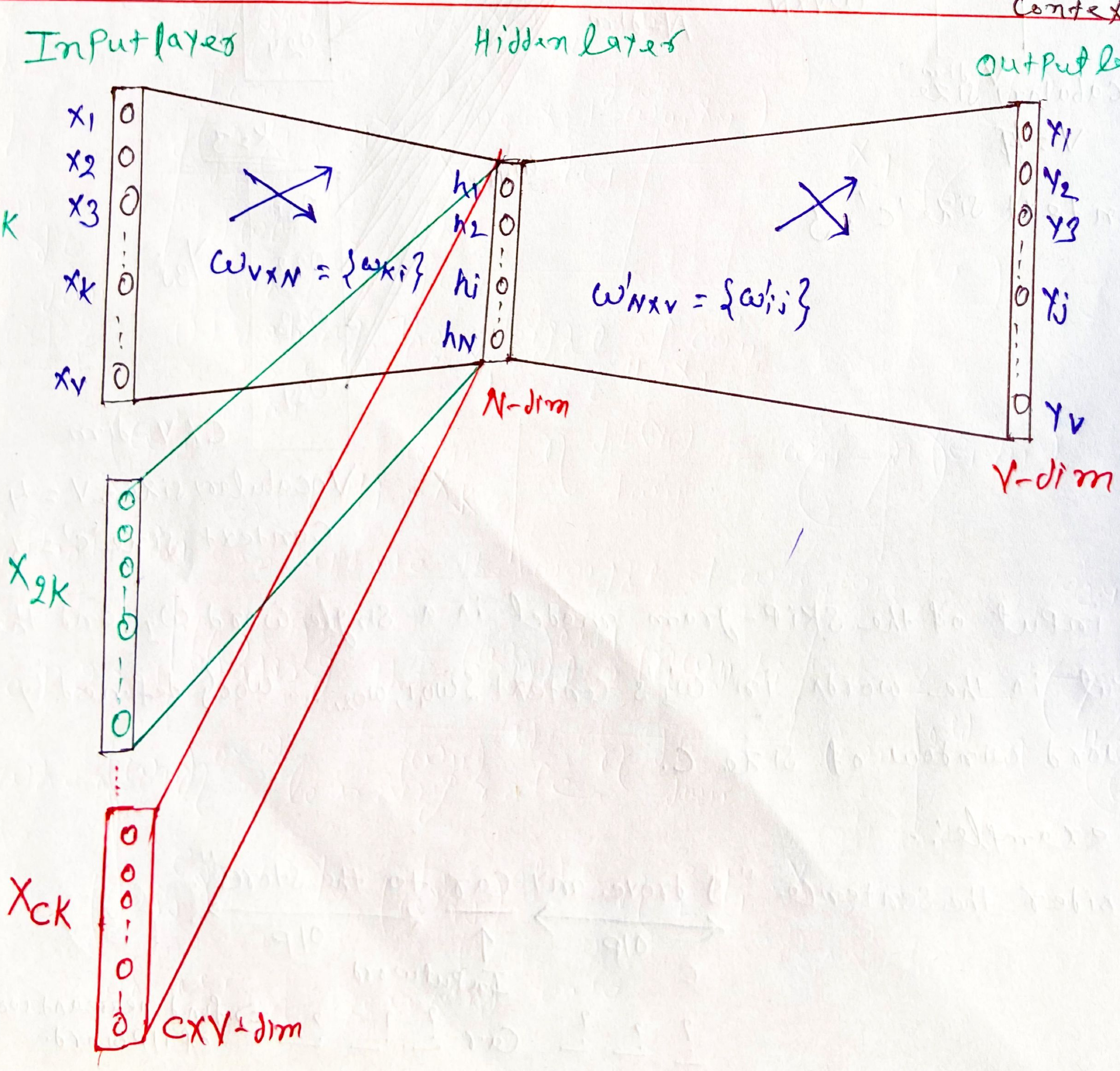


$(\omega)^{new} = (\omega)^{old} - \eta EH.X$

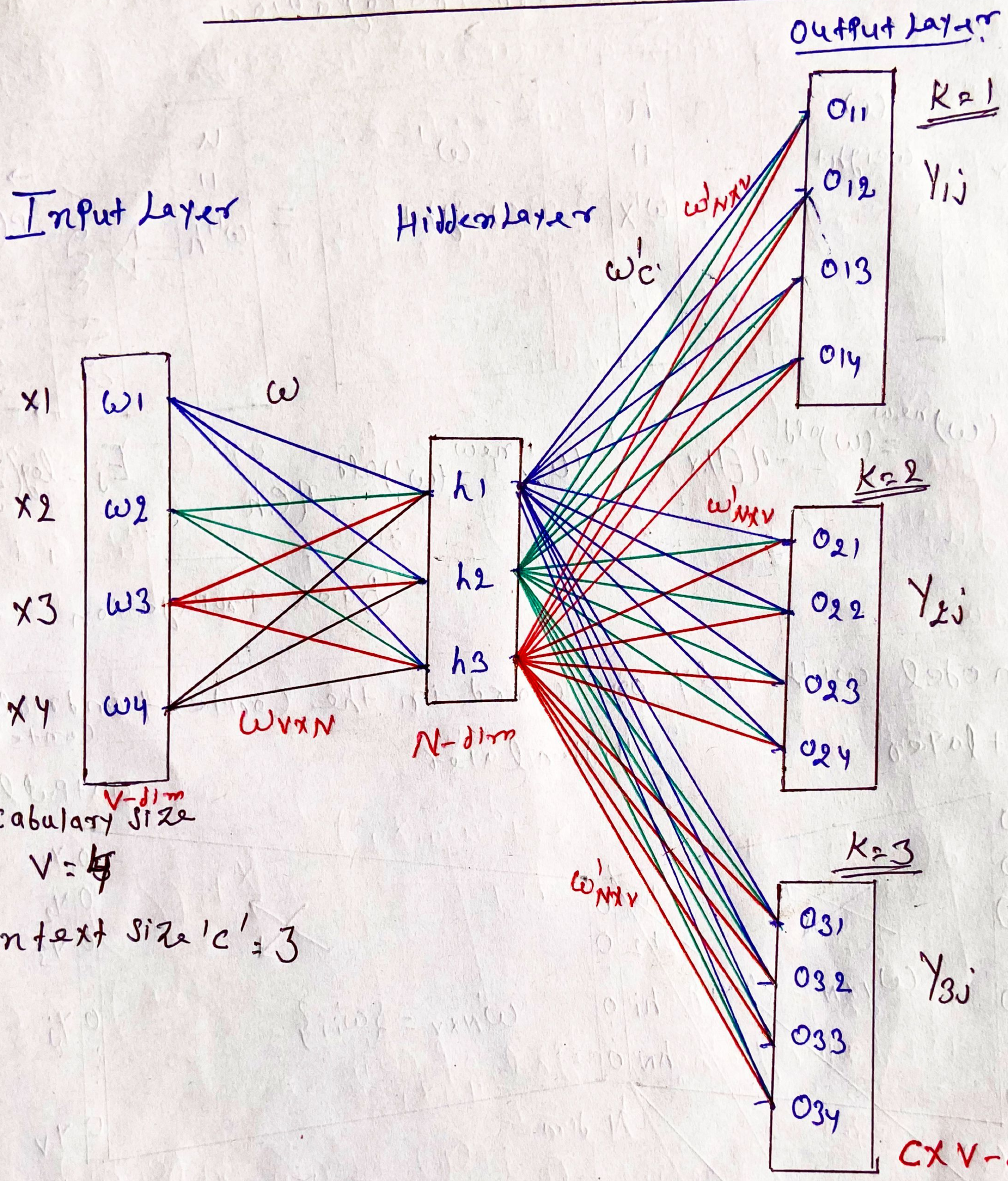
$(\omega')^{new} = (\omega')^{old} - \eta Eh$

← Back-Propagation

CBOW model with only one word in the context and multiple context



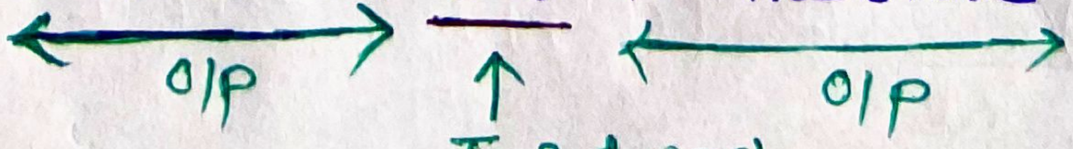
SKiP-Gram Model



The input of the SKip-gram model is a single word w_I and the output is the words in w_I's context $\{w_{O1}, w_{O2}, \ldots w_{Oc}\}$ defined by a word window of size C.

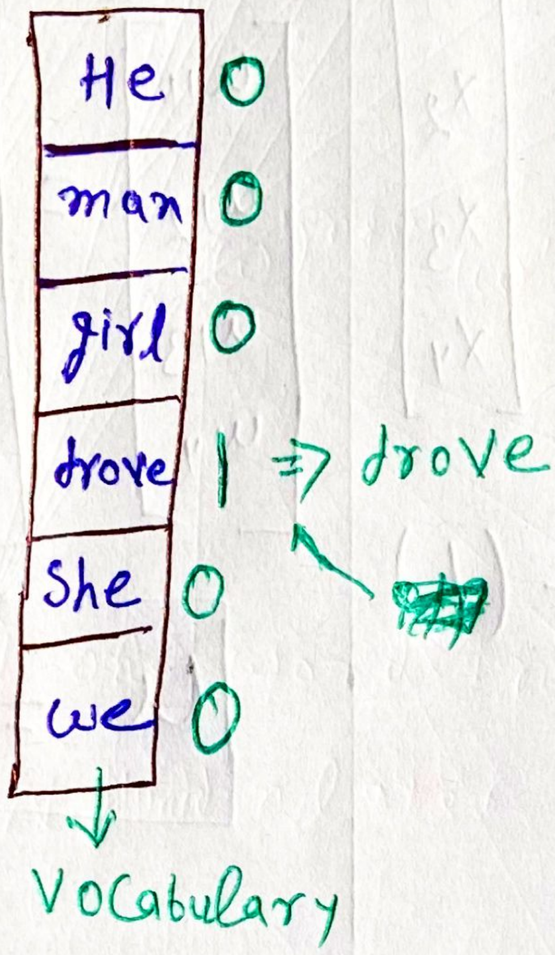
for example:-

Consider the sentence "I drove my Car to the store".



A Potential training instance could be the word "car" as an input and the words {"I", "drove", "my", "to", "the", "store"} as outputs.

All of these words are one-hot encoded meaning they are Vector of length V (the size of the Vocabulary) with a value 1 at the index ~~exam~~ corresponding to the word and Zeros(0) in all other indexes.



In this model X represents the one-hot encoded Vector corresponding to the input word in the training instance and $\{Y_1, Y_2 \ldots Y_C\}$ are the one-hot encoded Vectors corresponding to the output words in the training instance. The VXN matrix W is the weight matrix between the input layer and hidden layer whose ith row represents the weights corresponding to the ith word in the Vocabulary. Each output word Vector also has an associated NXV out matrix W'. There is also a hidden layer consisting of N nodes.

Weight b/w Input & hidden layer

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \\ w_{41} & w_{42} & w_{43} \end{bmatrix}$$

Weight b/w hidden & O/P layer.

$$W' = \begin{bmatrix} w'_{11} & w'_{12} & w'_{13} & w'_{14} \\ w'_{21} & w'_{22} & w'_{23} & w'_{24} \\ w'_{31} & w'_{32} & w'_{33} & w'_{34} \end{bmatrix}$$

Forward Propagation : weight Calculation

Input Layer to hidden Layer

$$h_1 = w_{11}x_1 + w_{21}x_2 + w_{31}x_3 + w_{41}x_4$$

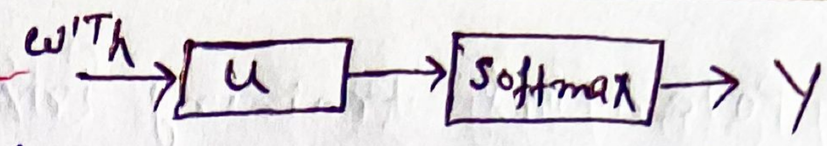
$$h_2 = w_{12}x_1 + w_{22}x_2 + w_{32}x_3 + w_{42}x_4$$

$$h_3 = w_{13}x_1 + w_{23}x_2 + w_{33}x_3 + w_{43}x_4$$

$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \\ w_{13} & w_{23} & w_{33} & w_{43} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$\boxed{h = w^T x} \quad \text{---(1)}$$

Hidden Layer to output Layer



$$u_{11} = w'_{11}h_1 + w'_{21}h_2 + w'_{31}h_3$$

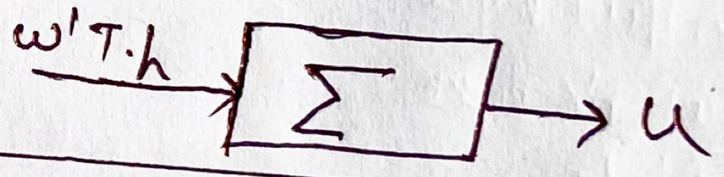
$$u_{12} = w'_{12}h_1 + w'_{22}h_2 + w'_{32}h_3$$

$$u_{13} = w'_{13}h_1 + w'_{23}h_2 + w'_{33}h_3$$

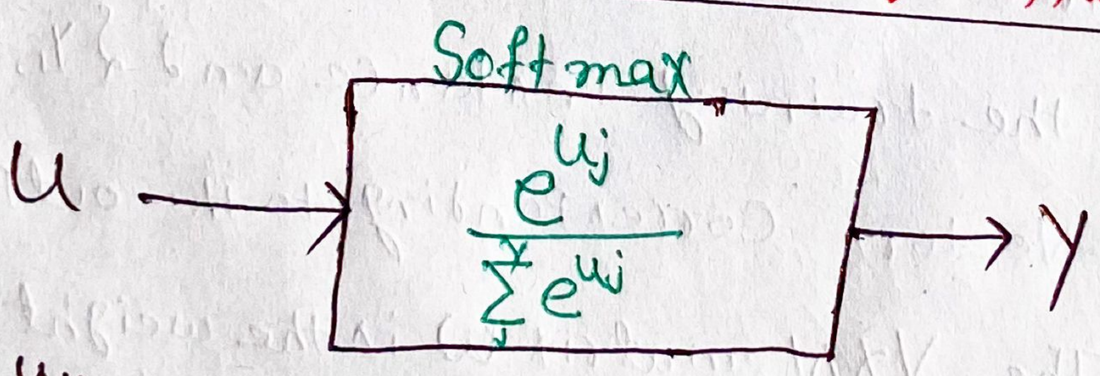
$$u_{14} = w'_{14}h_1 + w'_{24}h_2 + w'_{34}h_3$$

$$\begin{bmatrix} u_{11} \\ u_{12} \\ u_{13} \\ u_{14} \end{bmatrix} = \begin{bmatrix} w'_{11} & w'_{21} & w'_{31} \\ w'_{12} & w'_{22} & w'_{32} \\ w'_{13} & w'_{23} & w'_{33} \\ w'_{14} & w'_{24} & w'_{34} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix}$$

$$\boxed{u = w'^T h} \quad \text{---(2)}$$



Forward Propagation : Calculating Softmax output



$$y_{11} = \frac{e^{u_{11}}}{e^{u_{11}} + e^{u_{12}} + e^{u_{13}} + e^{u_{14}}}$$

$$y_{12} = \frac{e^{u_{12}}}{e^{u_{11}} + e^{u_{12}} + e^{u_{13}} + e^{u_{14}}}$$

The Softmax output for the word w_j with respect to the given output word w_I can be given by Conditional Probability, in terms of exponentials, i.e

$$P(w_j/w_I) = y_{w_j} = \frac{e^{u_j}}{\sum_{j=1}^{V} e^{u_i}} \quad \text{---(3)}$$

Error Calculation

Assumption: Let, $\{w_{1j}, w_{2j}, w_{3j} \ldots w_{cj}\}$ are actual output words, for the given single input word w_I. where, w_{ij} is the j^{th} word of 1^{st} Context window,

The Training objective is to maximize the conditional Probability of observing the actual output words, $\{w_{1j}, w_{2j}, w_{3j} \ldots w_{cj}\}$ given the Single input Context word w_I. Additionally, we want to minimize the error, so the loss function can be defined by Conditional Probability, in term of exponentials, i.e

$$E = -\log P(w_{1j}, w_{2j}, \ldots w_{cj} | w_I) = -\log \prod_{c=1}^{C} \left[\frac{e^{u_{cj}}}{e^{u_1} + e^{u_2} + \cdots + e^{u_J}} \right]$$

$$E \doteq -\log \prod_{c=1}^{C} \left[\frac{e^{u_{cj}}}{\sum_{j=1}^{J} e^{u_j}} \right]$$

$$\boxed{E = C \cdot \log \sum_{j=1}^{J} e^{u_j} - \sum_{c=1}^{C} u_{cj^*}} \quad \text{—— (4)}$$

where, C = total Context window

J = Vocabulary size

Now, taking derivative, w.r.t,

$$\frac{dE}{du_{cj}} = \frac{e^{u_{cj}}}{\sum_{j=1}^{J} e^{u_{cj}}} - Z_{cj} = Y_{cj} - Z_{cj} \quad \text{—— (5)}$$

where,

Y_{cj} = out score of j^{th} word of c^{th} Context window

$Z_{cj} = 1$, if the j^{th} word of c^{th} Context window in the actual output word, otherwise, we put $Z_{cj} = 0$

Back-Propagation : Output to Hidden Layer (update w')

In this back-propagation phase we, update the weight of all neurons from hidden layer to output layer. ($w'_{11}, w'_{12} \ldots$)

Step 1: Taking gradient of E' w.r.t, w'_{11}

$$\frac{dE(Y_{11})}{dw'_{11}} = \sum_{c=1}^{K} \frac{dE(Y_{11})}{du_{c1}} \times \frac{du_{c1}}{dw'_{11}}$$

$$\sum_{c=1}^{C} \frac{dE(Y_1)}{du_{c1}} = \sum_{c=1}^{C} (Y_{c1} - \overline{Z_{c1}}) = e_1$$

→ from eq^n (5)

$$\sum_{i=1}^{C} \frac{du_{c1}}{dw'_{11}} = \sum_{c=1}^{C} \frac{d(w'_{11} h_1 + w'_{21} h_2 + w'_{31} h_3)}{dw'_{11}} = h_1$$

$$\therefore \frac{dE(Y_{11})}{dw'_{11}} = \frac{dE(Y_1)}{du_1} \times \frac{du_1}{dw'_{11}} = e_1 h_1 \quad \text{—— (6)}$$

Step 2: Updating the weight of w'_{11}

$$(w'_{11})^{new} = (w'_{11})^{old} - \eta \frac{dE(Y_1)}{dw'_{11}}$$

$$(w'_{11})^{new} = (w'_{11})^{old} - \eta (e_1 \cdot h_1)$$

$$(w'_{ij})^{new} = (w'_{ij})^{old} - \eta (e_j \cdot h_i) \quad \text{—— (7)}$$

$$(w'_{ij})^{new} = (w'_{ij})^{old} - \eta \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot h_i \quad \text{— (7)}$$

$$\left[\frac{dE}{dw'_{ij}} = \sum_{c=1}^{C} \frac{dE}{du_{cj}} \times \frac{du_{cj}}{dw'_{ij}} = \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot h_i \right]$$

Back-Propagation: Hidden to Input Layer (update ω)

In this back-Propadation phase we update the weight of all neuron from hidden layer to output layer.

Step 1: Taking gradient of 'E' w.r.to h_1

$$\frac{dE}{dh_1} = \sum_{j=1}^{Y} \frac{dE}{du_{cj}} \times \frac{du_{cj}}{dh_1} = \sum_{j=1}^{Y} \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot \omega'_{1j}$$

Step 2: Calculating gradient of 'E', w.r.to ω_{11}

$$\frac{dE}{d\omega_{11}} = \frac{dE}{dh_1} \times \frac{dh_1}{d\omega_{11}} = \sum_{j=1}^{Y} \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot \omega'_{1j} \times (X_1)$$

Step 3: Updating the weight of ω_{11}

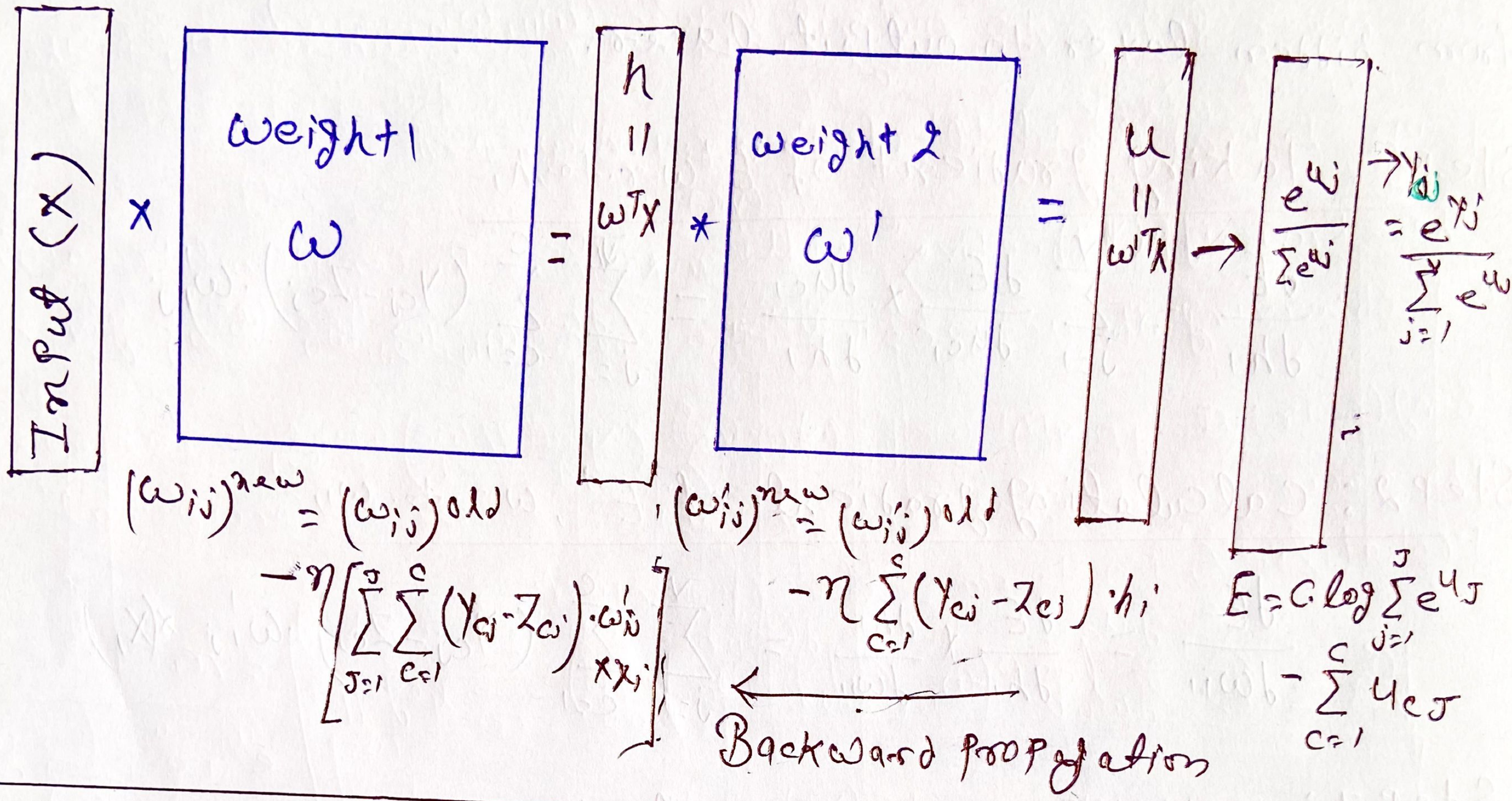
$$(\omega_{11})^{new} = \omega_{11} - \eta \frac{dE}{d\omega_{11}}$$

$$= \omega_{11} - \eta \left(\sum_{j=1}^{Y} \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot \omega'_{1j} \times (X_1) \right)$$

$$(\omega_{ij})^{new} = (\omega_{ij})^{old} - \eta \left(\sum_{j=1}^{Y} \sum_{c=1}^{C} (Y_{cj} - Z_{cj}) \cdot \omega'_{ij} \times (X_i) \right) \quad -(8)$$

Model

forward Propagation →



Skip-gram