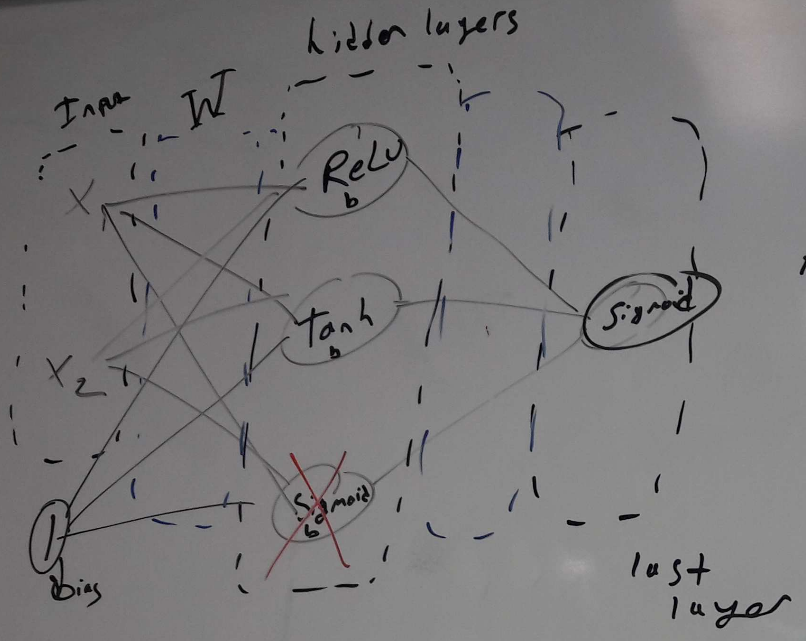


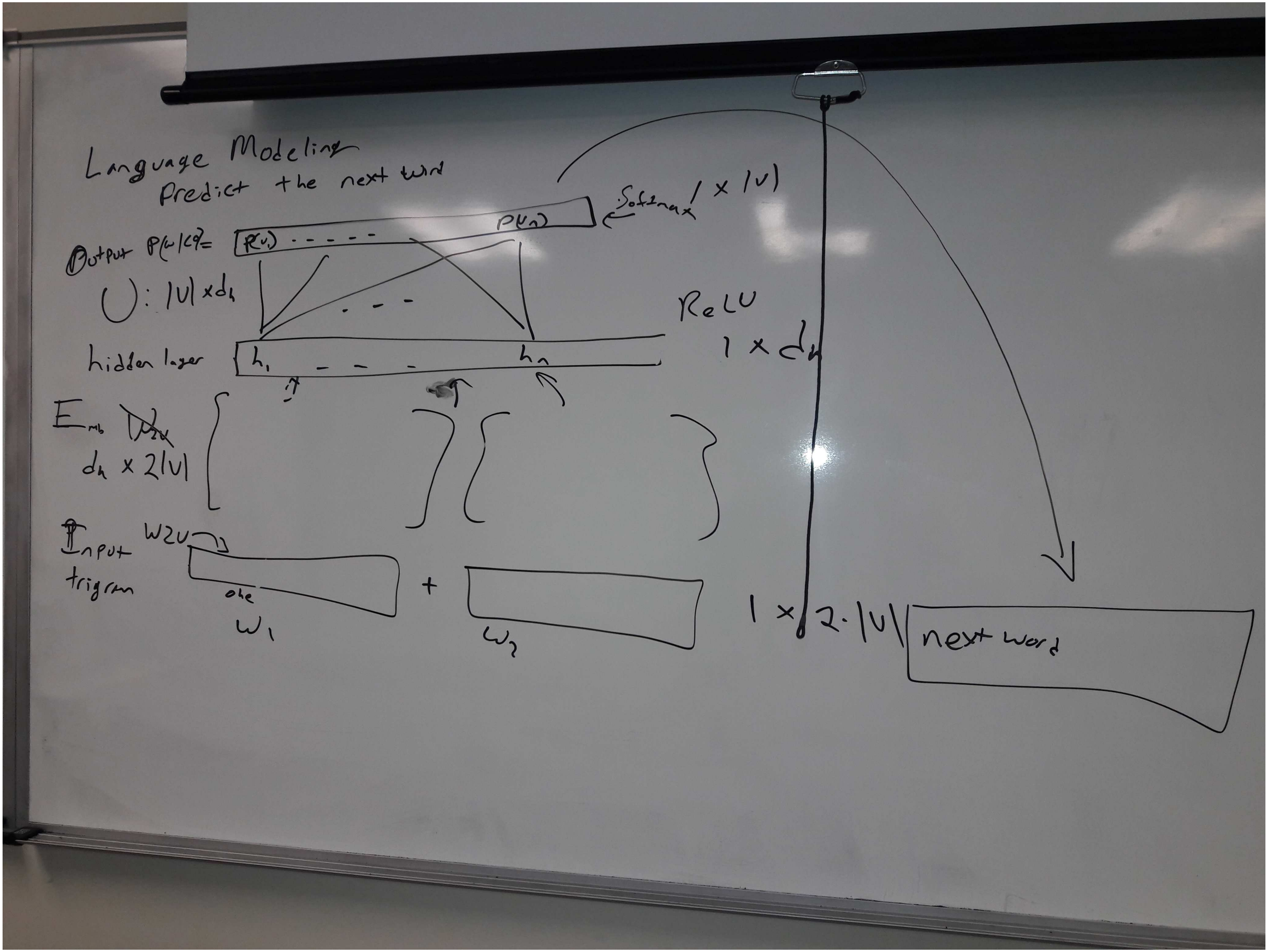
Alt last layer / softmax $\begin{bmatrix} - \\ - \end{bmatrix}$

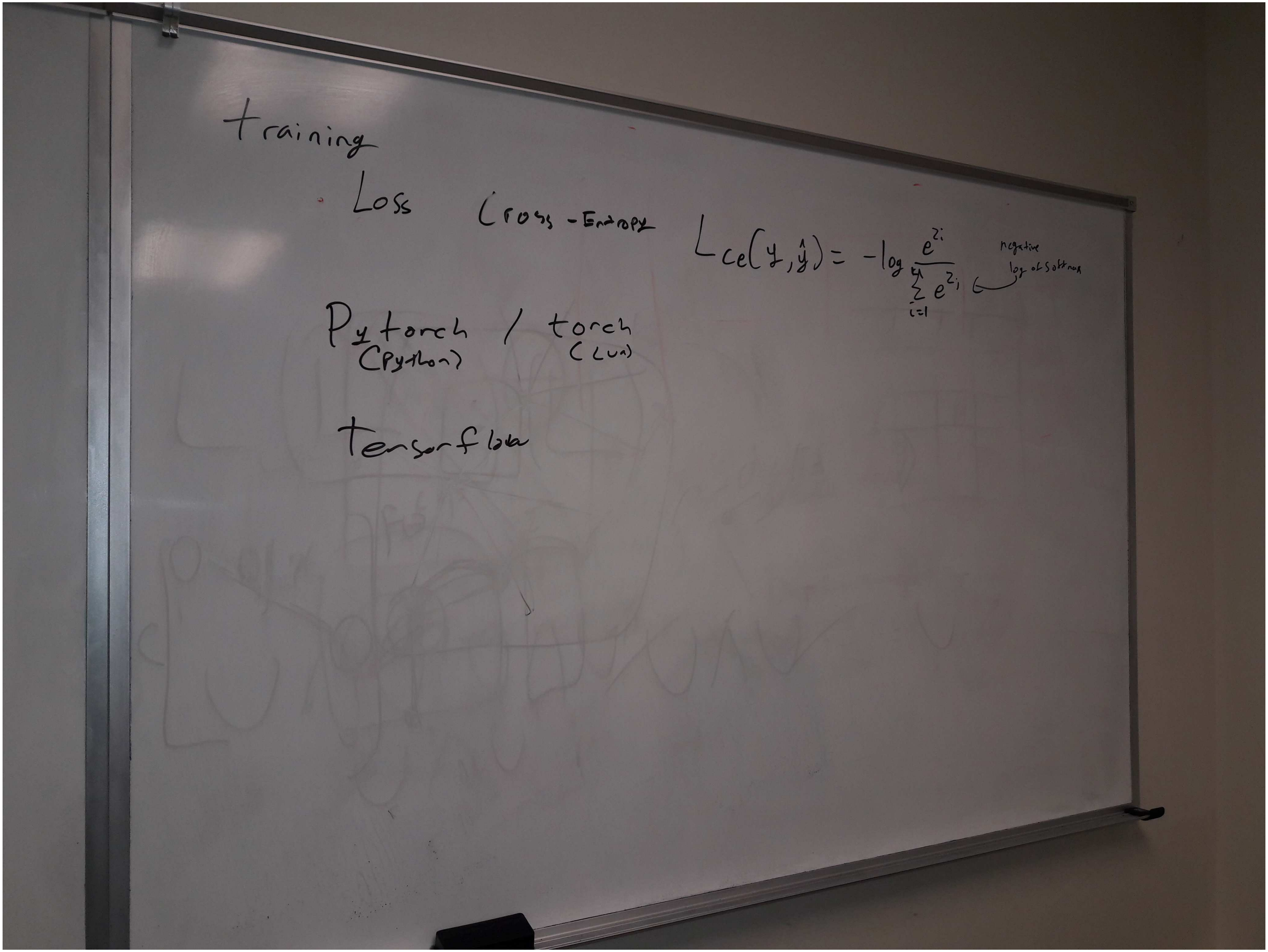
$$\text{Relu}(W_{r1} \cdot x_1 + w_{r2} x_2 + b)$$

2-layer

$$\underset{3\times1}{h} = \partial\left(\underset{3\times2}{W} \cdot \underset{2\times1}{x} + b\right)$$

$$\hat{y} = \text{Sigmoid}(h)$$


Language Modeling
Predict the next word
Output P(w|c)=
P(v1) - - - - - -
P(vn)
Softmax
1 × |V|
U: |V| × dh
ReLU
1 × dh
hidden layer
h1 - - -
hn
Emb ~~W2u~~
dh × 2|V|
Input
trigram
W2V
one
w1
+
w2
1 × 2·|V|
next word



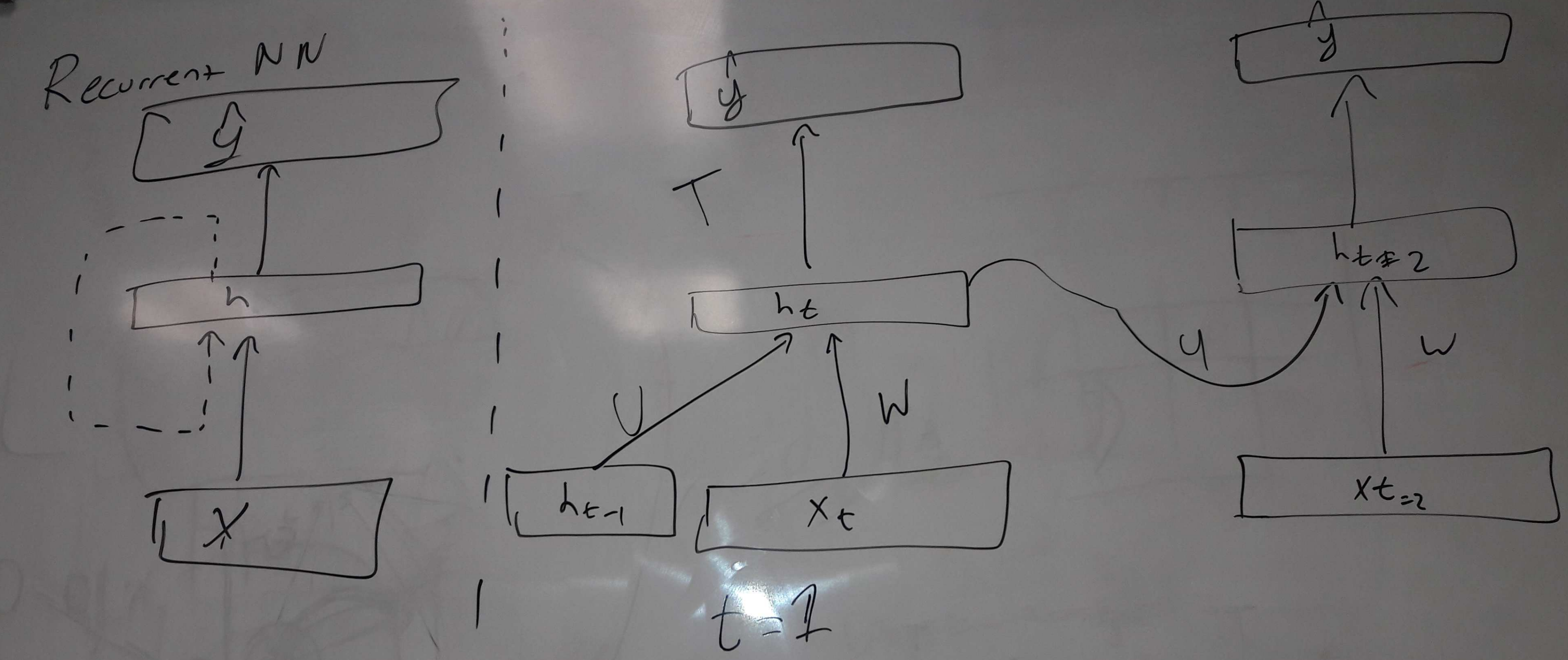
training

Loss Cross-Entropy $L_{ce}(y,\hat{y}) = -\log \frac{e^{z_i}}{\sum_{i=1}^{k} e^{z_i}}$ negative log of softmax

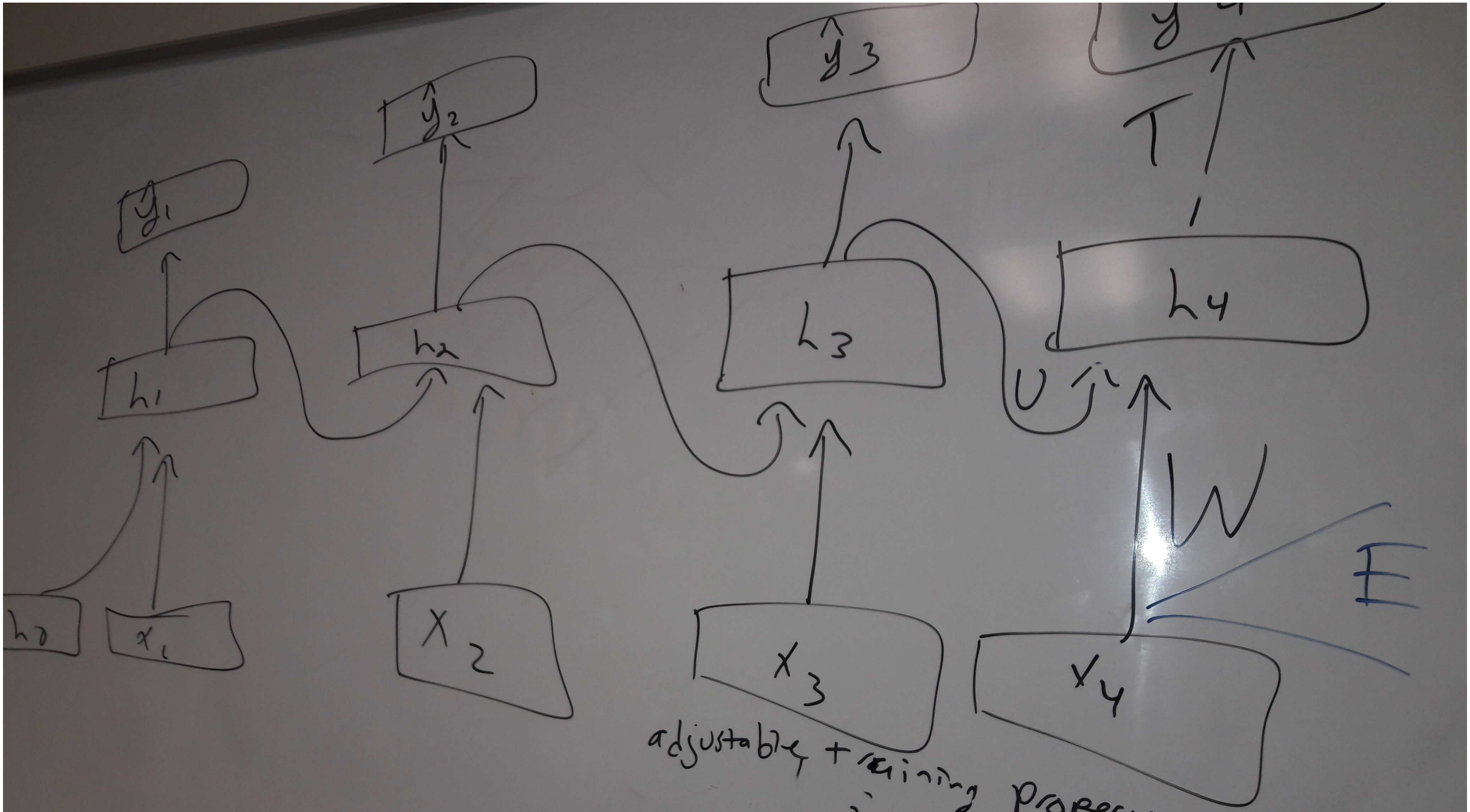
Pytorch (Python) / torch (Lua)

Tensorflow

Recurrent NN



w_1	w_2	w_3
tag_1	tag_2	tag_1



adjustable + training properties

Length -

Batching

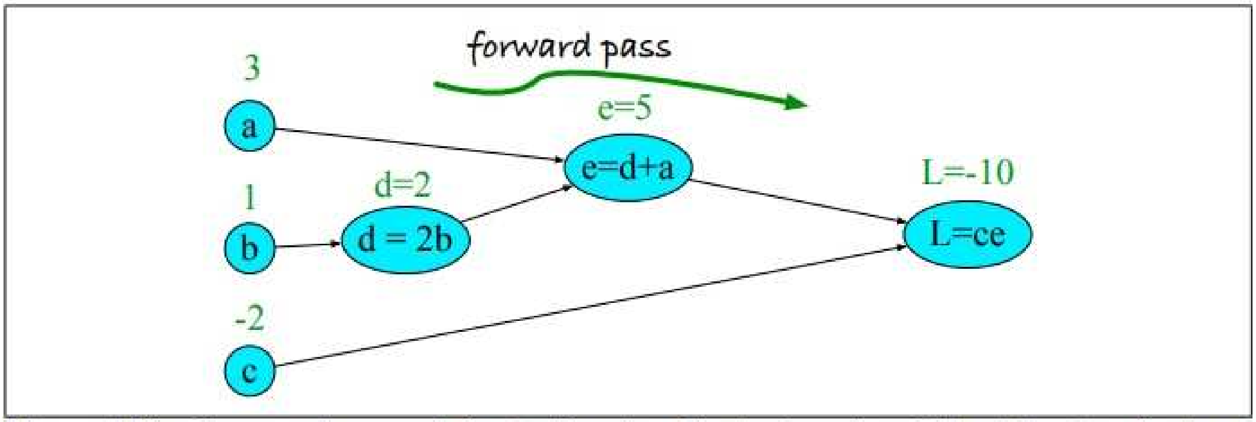


Figure 7.9 Computation graph for the function $L(a,b,c) = c(a+2b)$, with values for input nodes $a = 3$, $b = 1$, $c = -2$, showing the forward pass computation of L.

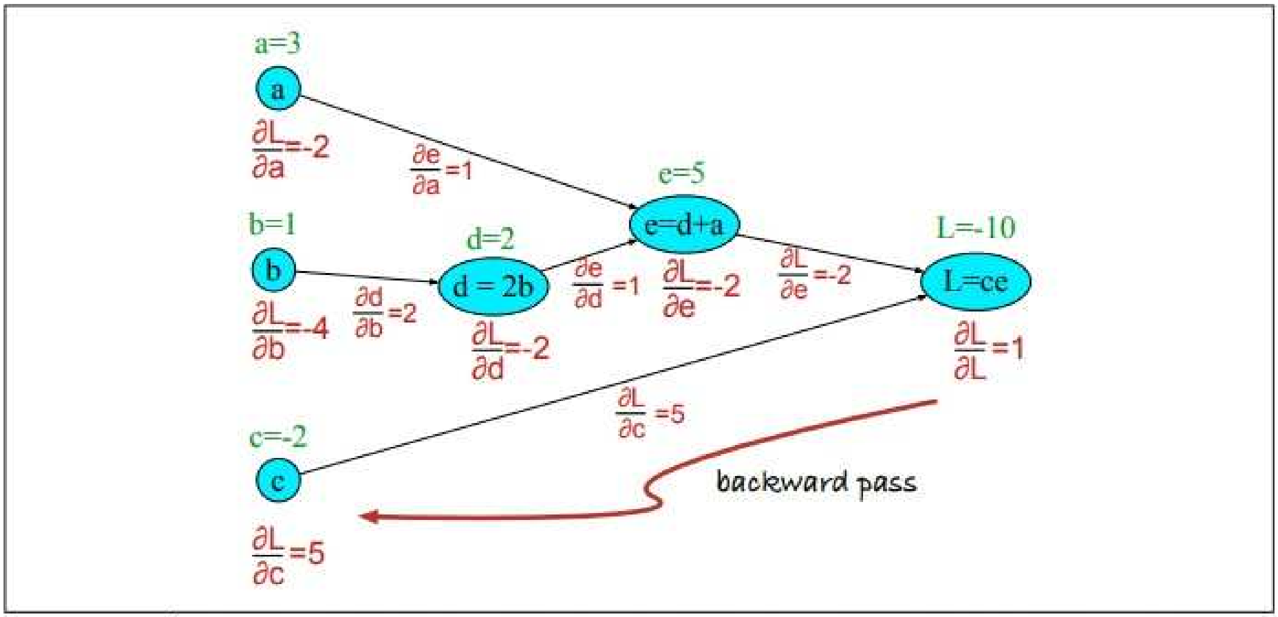


Figure 7.10 Computation graph for the function $L(a,b,c) = c(a+2b)$, showing the backward pass computation of $\frac{\partial L}{\partial a}$, $\frac{\partial L}{\partial b}$, and $\frac{\partial L}{\partial c}$.

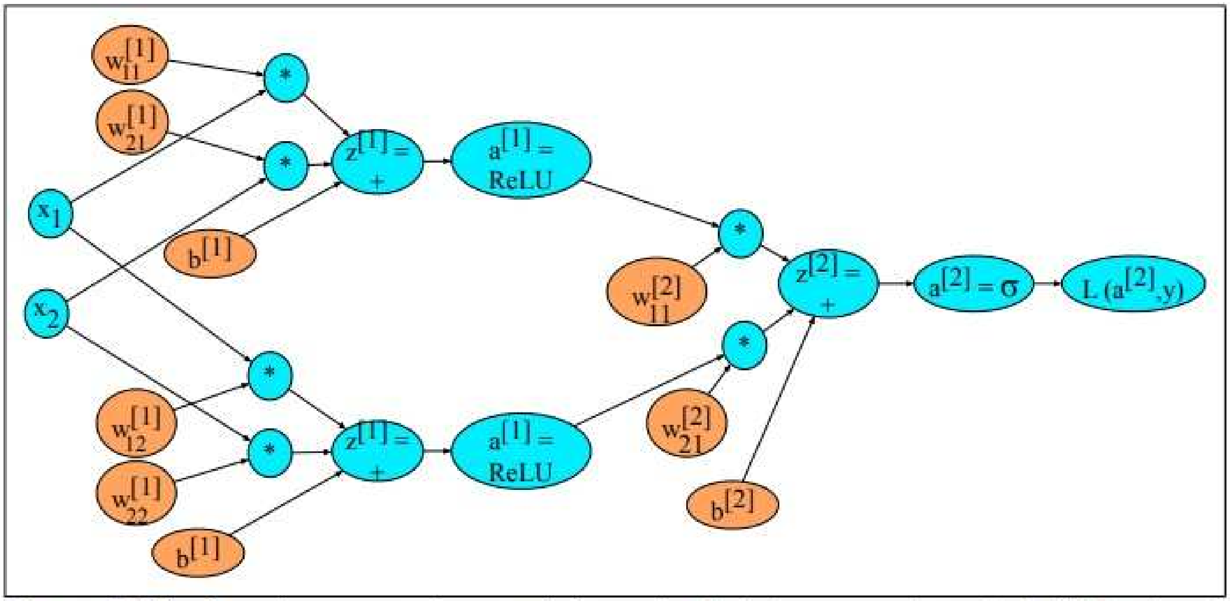


Figure 7.11 Sample computation graph for a simple 2-layer neural net (= 1 hidden layer) with two input dimensions and 2 hidden dimensions.