https://github.com/TAMIDSpiyalong/Gen-AI

Lecture 1: 9:00-9:40

Yalong Pi (Texas A&M Institute of Data Science)

Address: Office 221C, John R. Blocker Building

Email: piyalong@tamu.edu

- B.S., Mechanical Engineering, 2007-2011
- M.S., Civil Engineering, 2011-2013
- Ph.D., Architecture, 2017-2020

- Research Scientist, 2020-present
- Architect, 2016-2017
- Project manager, 2013-2016

Agenda

- **Part 1: Background and Theory on Generative AI**
- Fundamentals of machine learning 9:00 – 10:00
- Tokenization and word embedding 10:00 – 10:30
- Transformers for Language Models 10:30 – 12:00
- **Part 2: Applications and Hands-On Exercises**
- Prompt Engineering
- Generative AI Applications
- Evolution of Generative AI and Future Directions

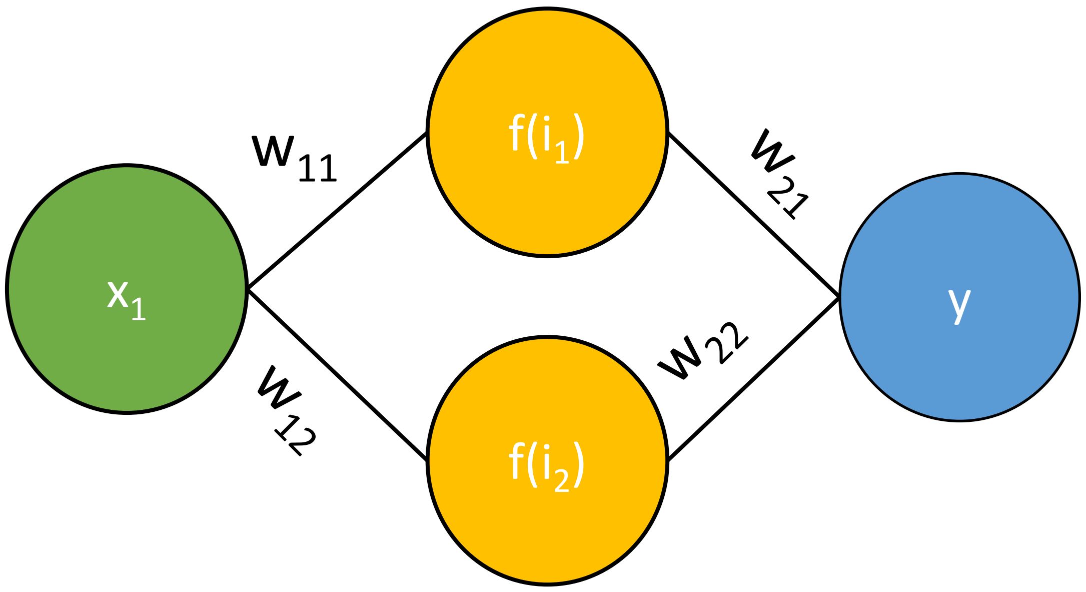
Fundamentals of machine learning

NLP

CV

alfa fold

recomendation



Back Propagation

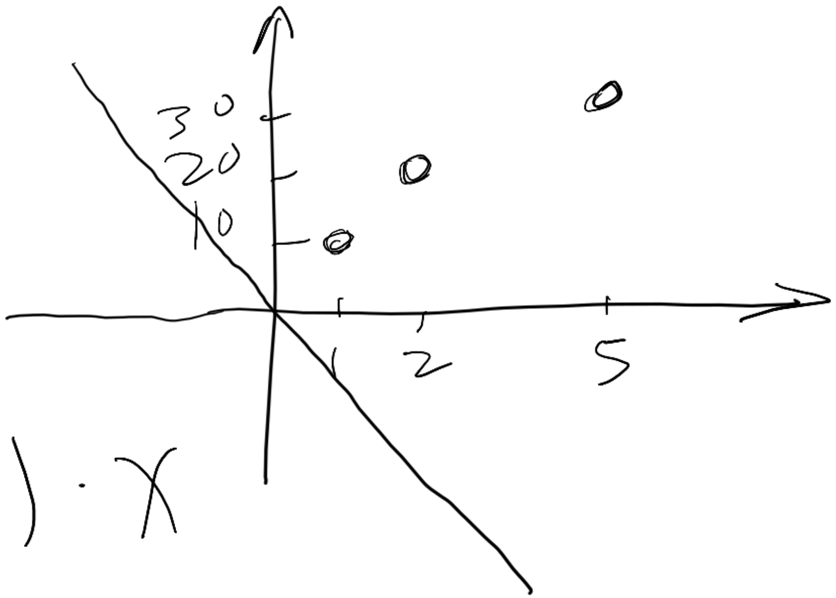
x	y
1	10
2	20
5	30

$$y_{pred} = w_{21} * x_1 * w_{11} + w_{22} * x_1 * w_{12}$$

$$Loss = (y_{pred} - y_{true})^2$$

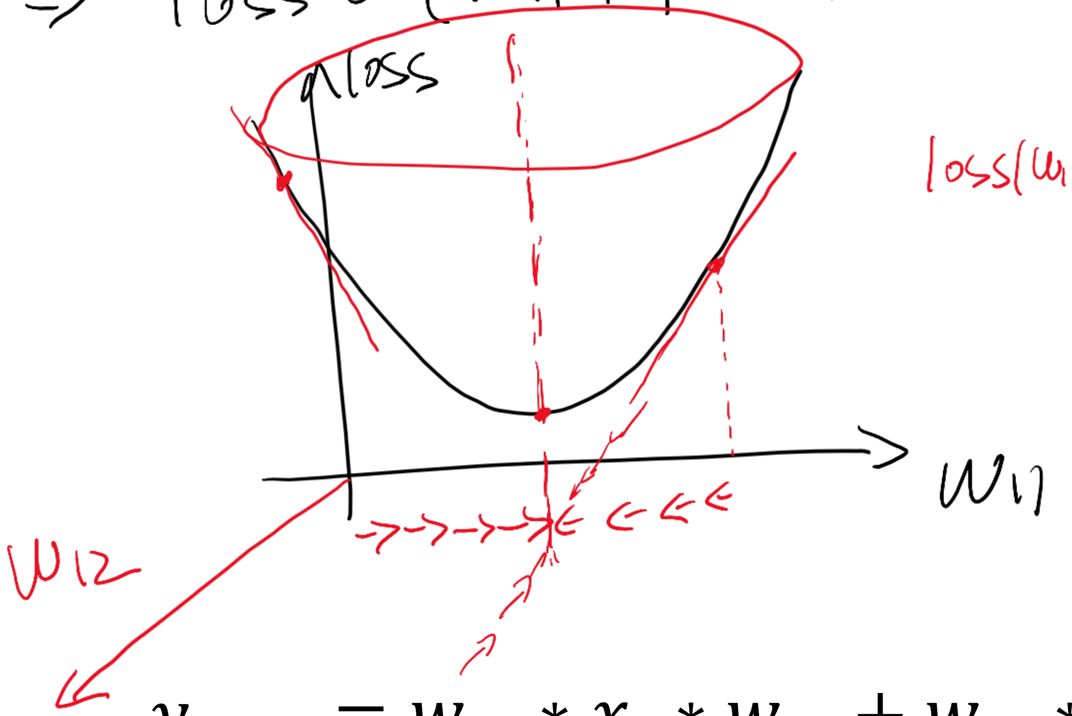
$$W_n' = Wn - LR\ (\partial Loss/\partial W_n)$$

$$y = (W_{11} \cdot W_{21} + W_{22} \cdot W_{12}) \cdot X$$



$$x = 1 = W_{21} = W_{22} = W_{12}$$

$$\Rightarrow loss = (W_{11} + 1 - 10)^2 = (W_{11} - 9)^2 = W_{11}^2 - 18W_{11} + 81$$



$$loss(W_{11})' = \frac{\Delta loss}{\Delta W_{11}} = \frac{\partial loss}{\partial W_{11}} = 2W_{11} - 18 = \frac{loss(W_{11} + \Delta W_{11})}{\Delta W_{11}}$$

$$W_{11}' = W_{11} - \frac{\partial loss}{\partial W_{11}} \cdot LR$$

$$W_{12}' = W_{12} - \frac{\partial loss}{\partial W_{11}}$$

$$y_{pred} = w_{21} * x_1 * w_{11} + w_{22} * x_1 * w_{12}$$

$$Loss = (y_{pred} - y_{true})^2$$

$$W_n' = Wn - LR\ (\partial Loss/\partial W_n)$$

x	y
1	10 ✓
2	20 ✓
5	30 ✓

$$\begin{bmatrix} w_{11}' \\ w_{12}' \\ w_{21}' \\ w_{22}' \end{bmatrix} = \begin{bmatrix} w_{11} \\ w_{12} \\ w_{21} \\ w_{22} \end{bmatrix} - \begin{bmatrix} \frac{\partial loss}{\partial w_{11}} \\ \frac{\partial loss}{\partial w_{12}} \\ \frac{\partial loss}{\partial w_{21}} \\ \frac{\partial loss}{\partial w_{22}} \end{bmatrix} \cdot LR$$

SGD adam

$$\vec{W}' = \vec{W} - \nabla \cdot \underbrace{LR}_{0.001}$$

batch

$$y_{pred} = w_{21} * x_1 * w_{11} + w_{22} * x_1 * w_{12}$$

$$Loss = (y_{pred} - y_{true})^2$$

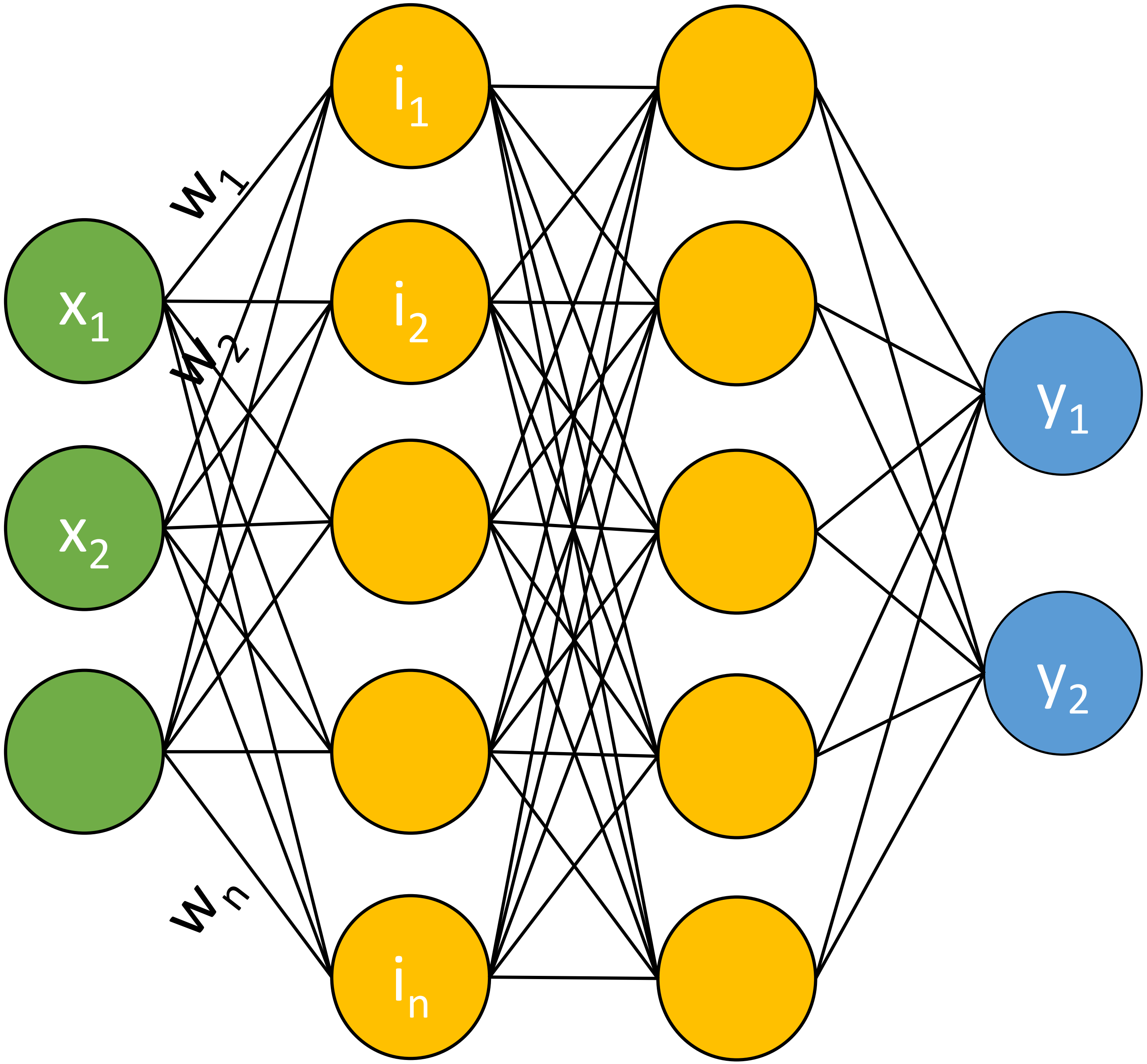
$$W_n' = Wn - LR\,(\partial Loss/\partial W_n)$$

x	y	
1	10	∇_1
2	20	∇_2
5	30	∇_3

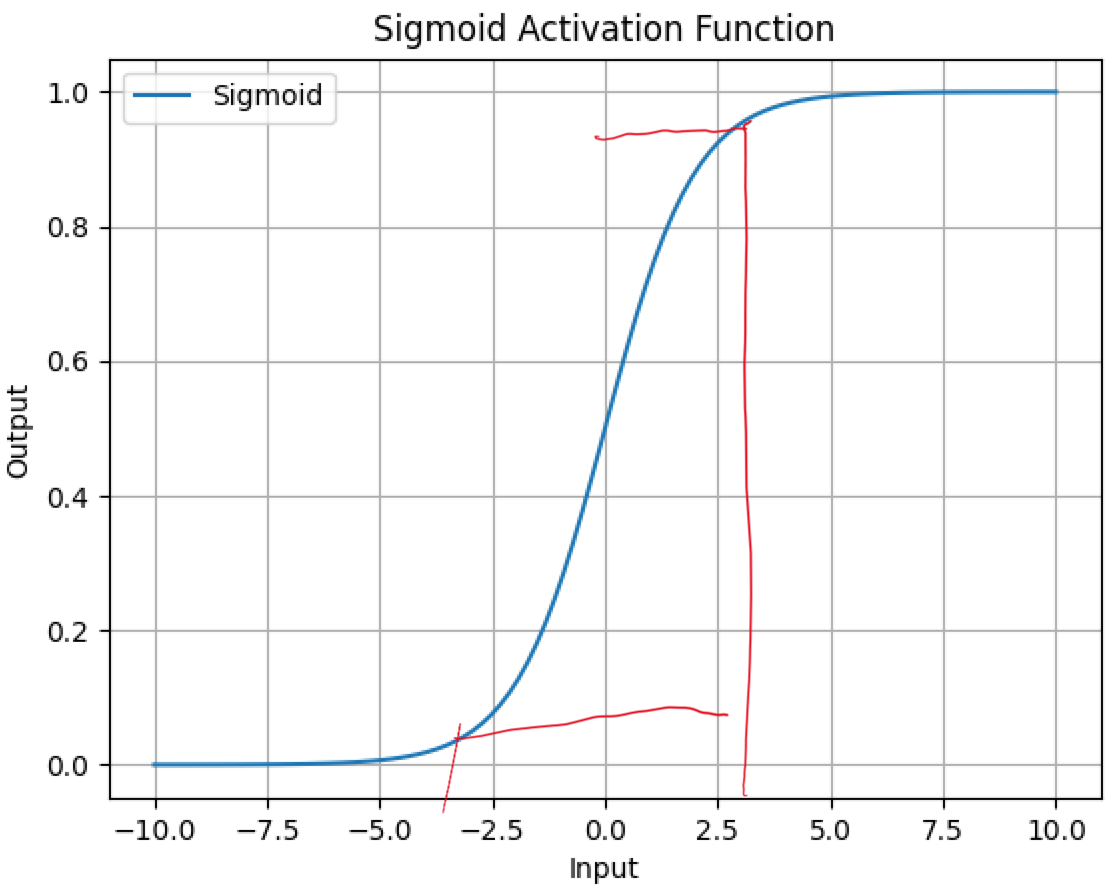
For each connection:

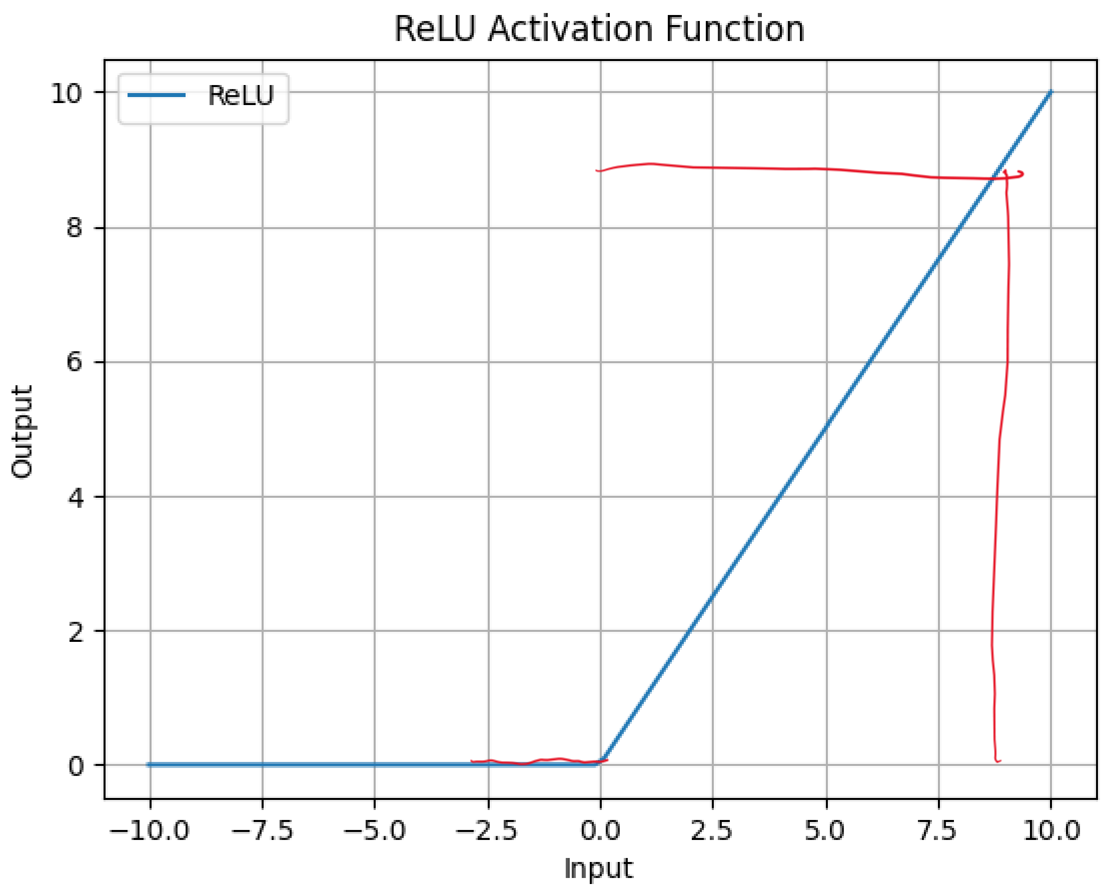
$$I_n = f(\sum_n x_n w_n + b)$$

- f is the activation function
- w_n is the weight
- b is the bias.
- A DNN has millions of weights and biases



Activation Functions

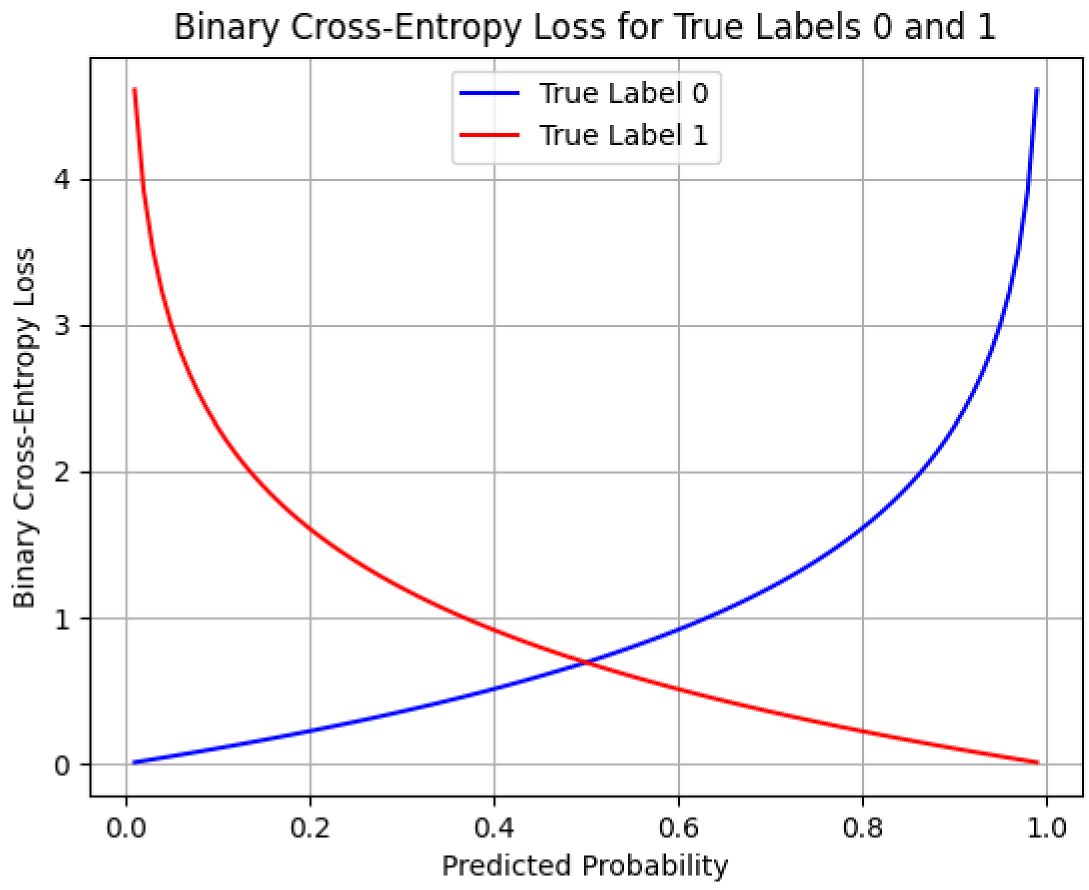




Binary Cross Entropy

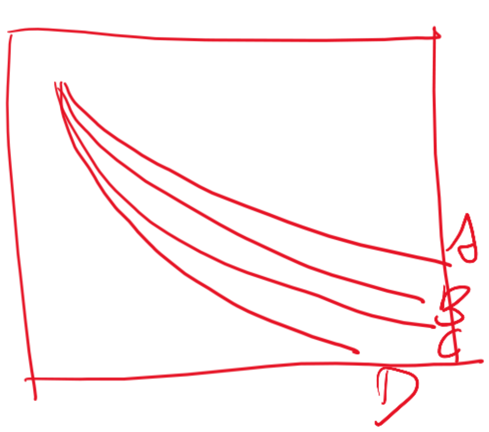
$$L(y, \hat{y}) = -[\, y * \log(\hat{y}) + (1 - y) * \log(1 - \hat{y}) \,]$$

Where:
- y is the true label (0 or 1)
- ŷ is the predicted probability (between 0 and 1)
- log can be ln



Softmax

CE



$$\sigma(Z)_i = \exp(z_i) \,/\, \Sigma(\exp(z_j))$$

Where:

- z_i is the i input score (logits)
- Σ is the sum over all input scores (logits)
- $\sigma(z)_i$ is the probability assigned to class i

not after softmax

Example:

[-0.37, -1.06, -0.07, -1.47, -0.90] **->** [0.265, 0.133, 0.358, 0.088, 0.155]

A B C D E

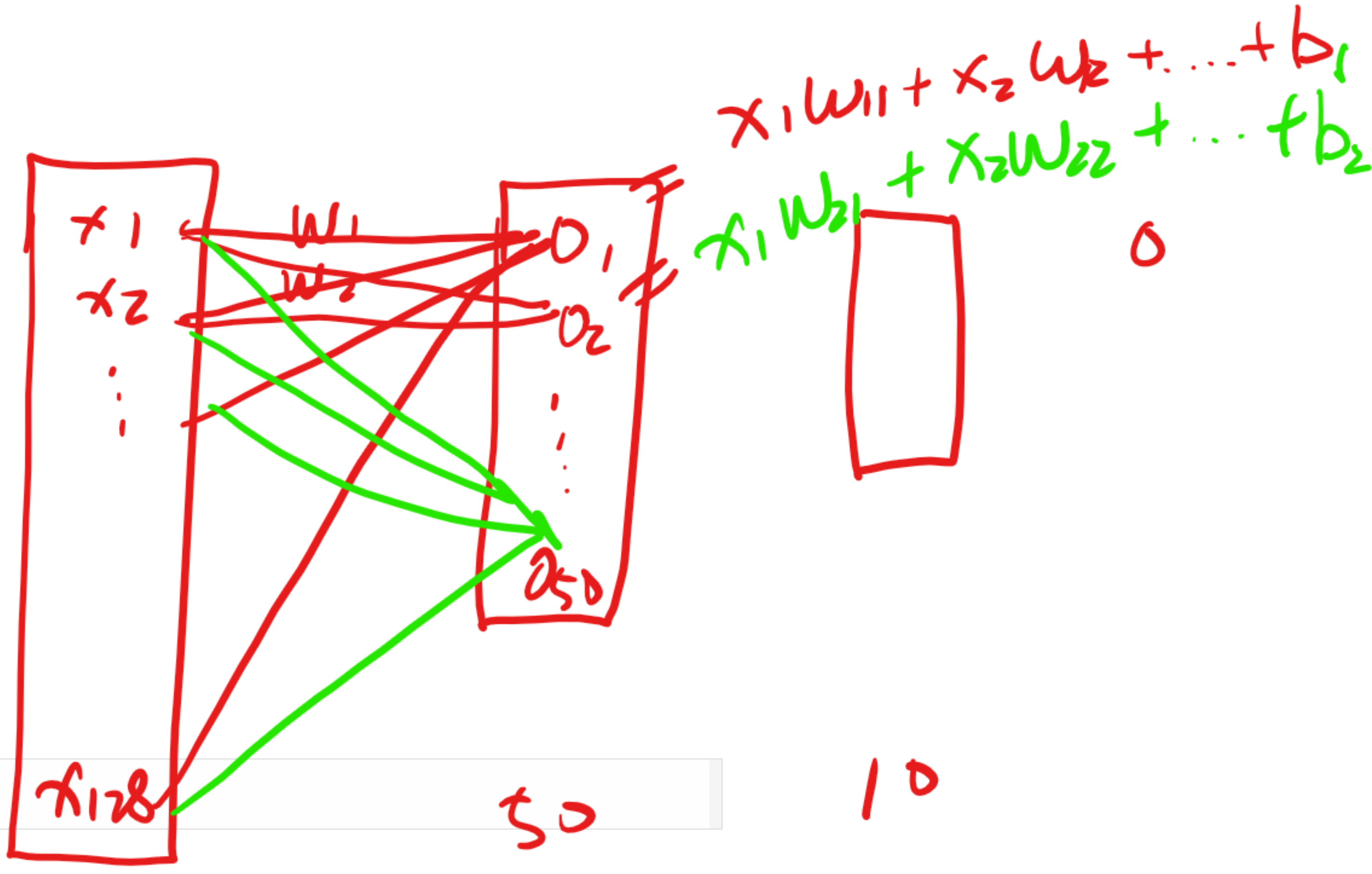
Prediction\Ground Truth	Positive	Negative
Positive	TP	FP
Negative	FN	TN

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

```
-4.37806658e-02, -9.67954174e-02, -1.28794938e-01,
 1.45295113e-01,  3.34722281e-01, -2.49092355e-02,
-7.21996874e-02,  7.60333985e-02,  1.09788142e-01,
-5.91083243e-02,  1.77998230e-01,  1.05147131e-01,
 2.73706466e-01,  1.63680017e-01,  2.92986393e-01,
 1.62288636e-01,  1.92936987e-01, -7.25108087e-02,
 1.48647577e-01,  1.20697133e-01,  1.75806686e-01,
-8.22802186e-02,  3.19161601e-02,  9.75683853e-02,
-2.27390900e-01, -1.89130962e-01, -7.75573701e-02,
 7.51652941e-02, -9.91581455e-02,  9.62962583e-03,
 6.42622411e-02, -1.50064066e-01,  1.14945382e-01,
 5.34672337e-03,  1.96428418e-01, -2.10412573e-02,
 5.44419959e-02, -3.22782189e-01,  5.69203123e-03,
-1.00528084e-01, -7.29111880e-02, -1.84138656e-01,
-1.51076904e-02, -6.30587935e-02]], dtype=float32)
```



In [ ]:
```
hub_layer(train_examples[:3]).shape
```

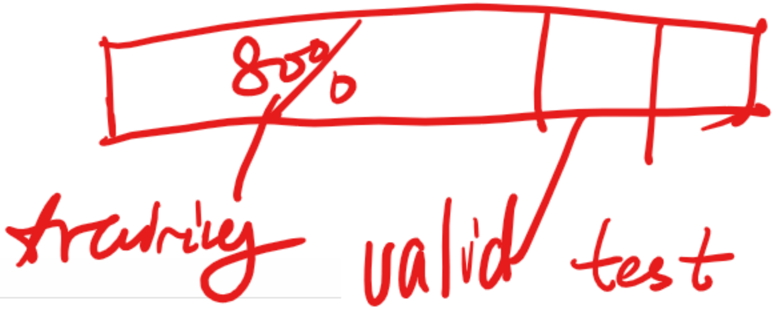
Out[ ]:
```
TensorShape([3, 128])
```

$128 \times 50 + 50 = 6450$

Let's now build the full model:

In [ ]:
```
model = tf.keras.Sequential()
model.add(tf.keras.layers.Dense(50, activation='relu',input_dim=128))
model.add(tf.keras.layers.Dense(10, activation='relu'))
model.add(tf.keras.layers.Dense(1))
```

```
plt.show()
```



early stop

save the best

data augmentation

regularization

drop out

Tokenization and word embedding

Tokenization

“I want pizza”

“我想要披萨”

“ピザが欲しいです”

“Eu quero pizza”

“أريد بيتزا”

“मुझे पिज़्ज़ा चाहिए”

“Quiero pizza”

“피자가 먹고 싶어요”

pizzas

wanted

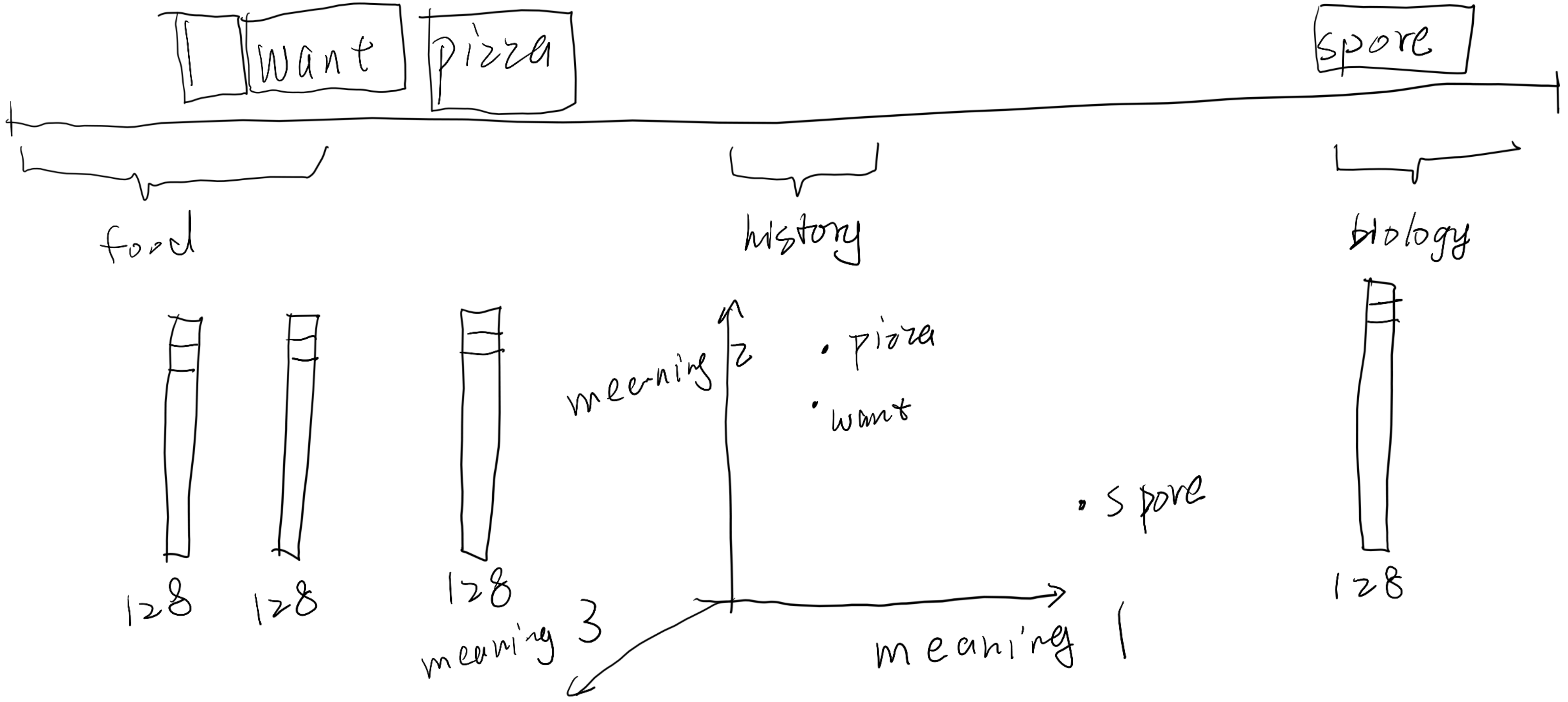
token vocabulary

ASCII

40k-45k

50k GPT

Word2Vec: Skig-gram and Negative Sampling



maybe I $\boxed{\text{want}}$ pizza from

target context

spore

want spore

want pizza

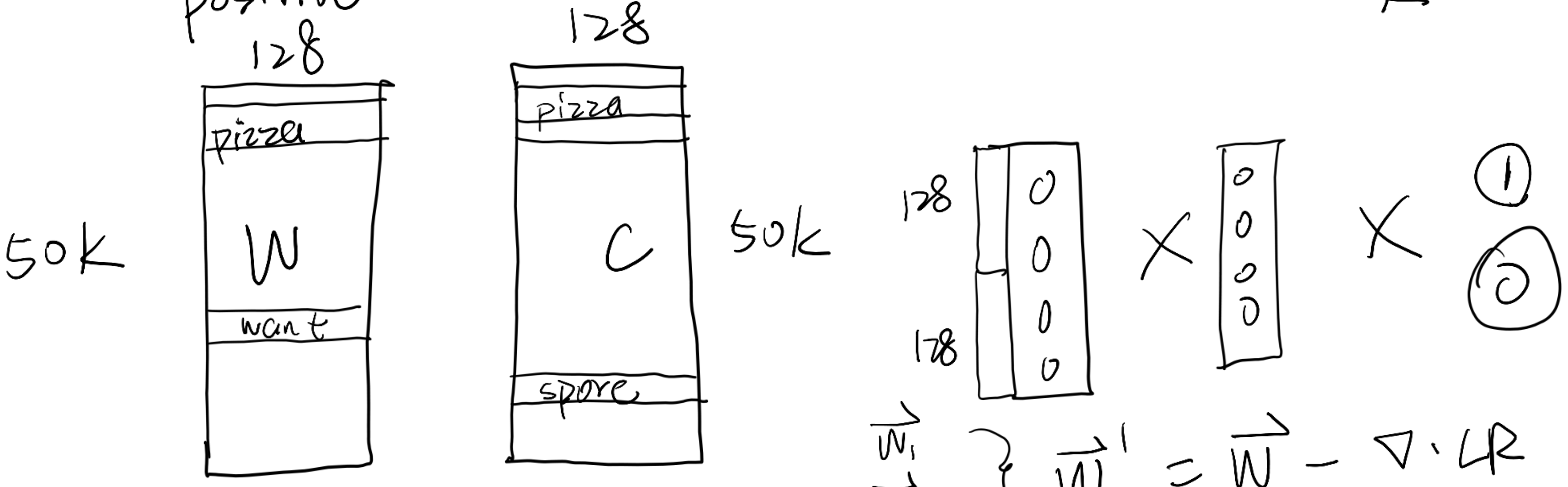
want I

want from

want mayb

positive

K negative

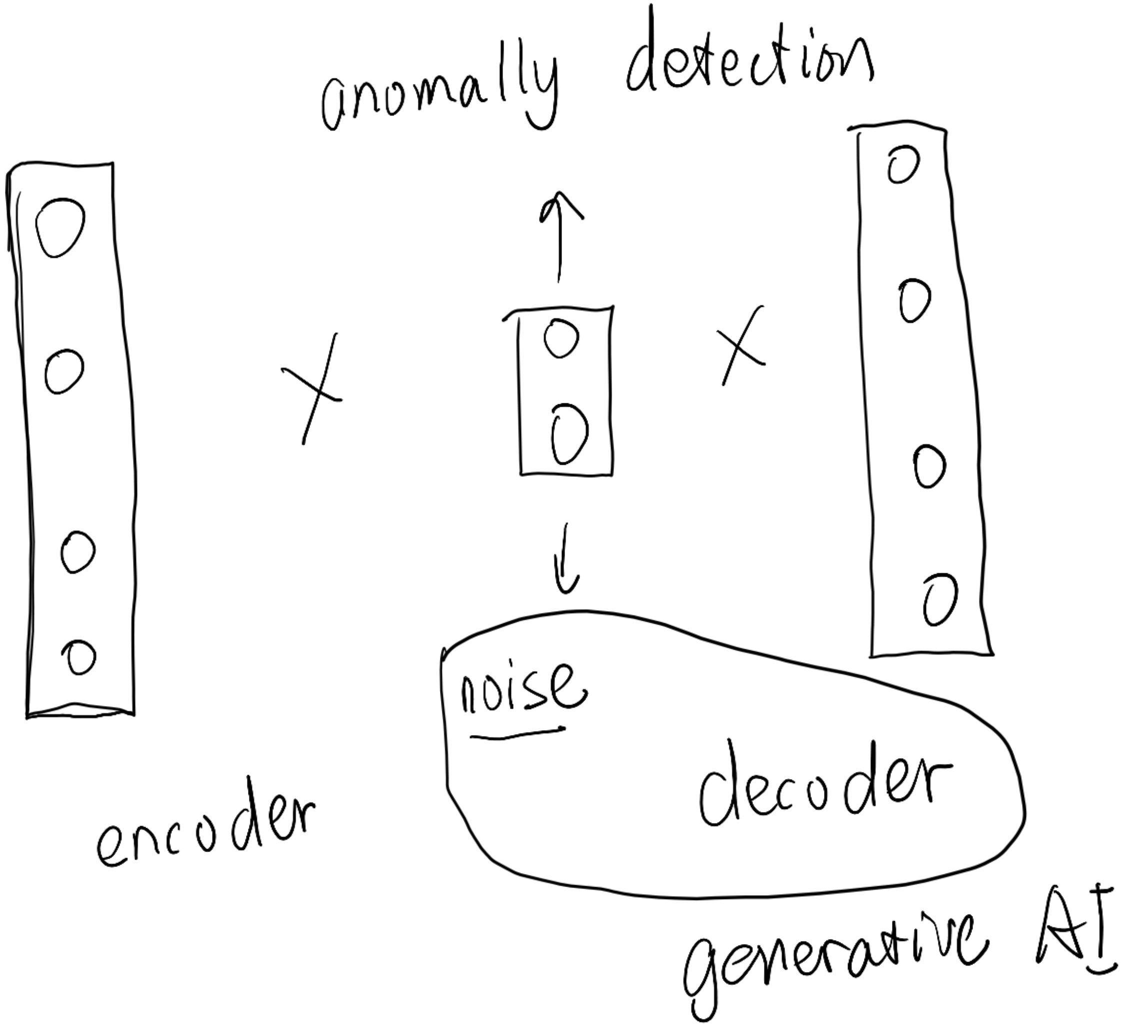


$\vec{w}_1$, $\vec{w}_2$ $\}$ $\vec{W}' = \vec{W} - \nabla \cdot LR$ ②

Transformers for Language Models of machine learning

Transformer

- GPT: Generative Pre-trained Transformer
- Vision Transformer: DETR
- Attention mechanism



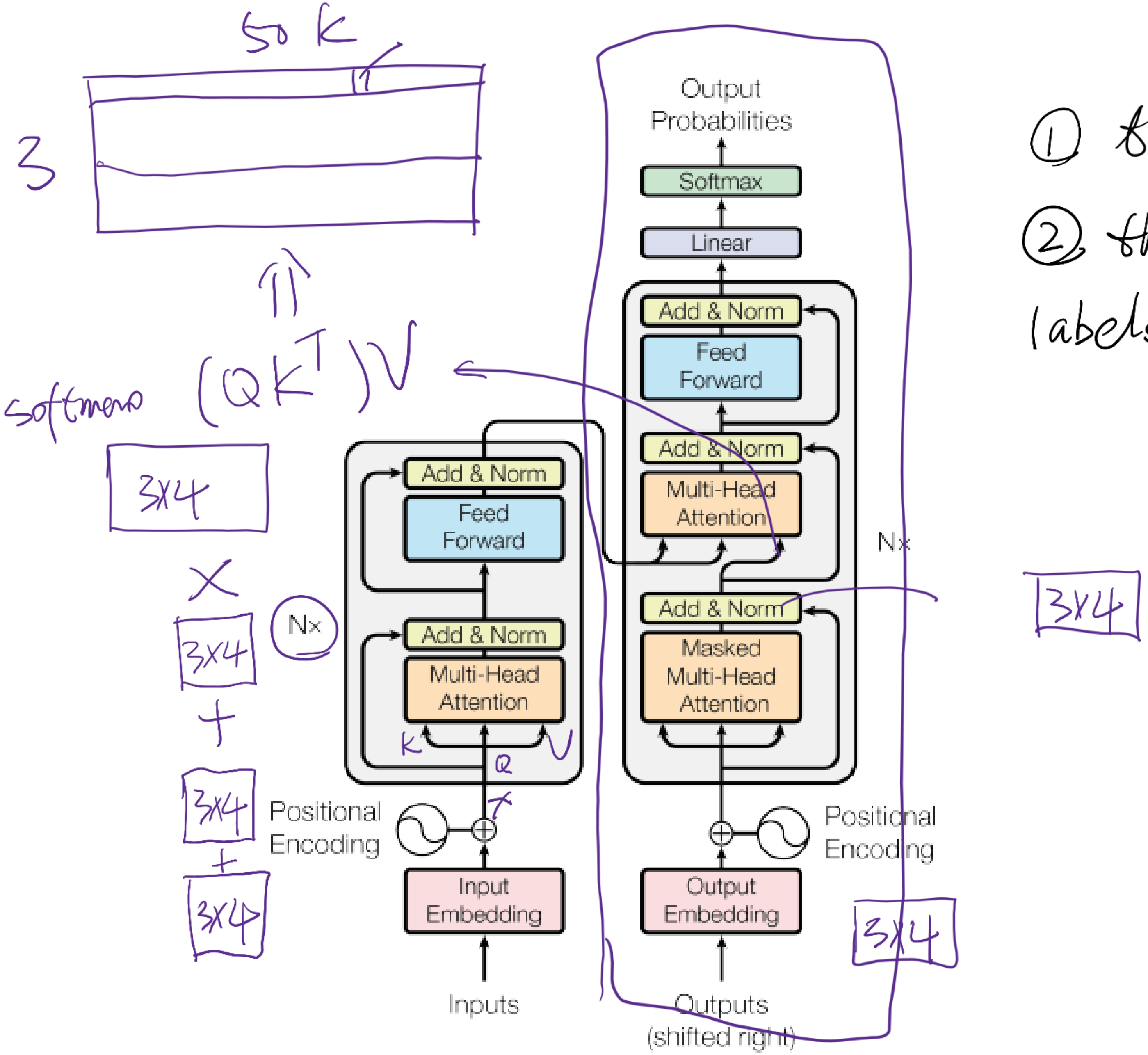
mse

CE

combination

unsupervised

no ground truth

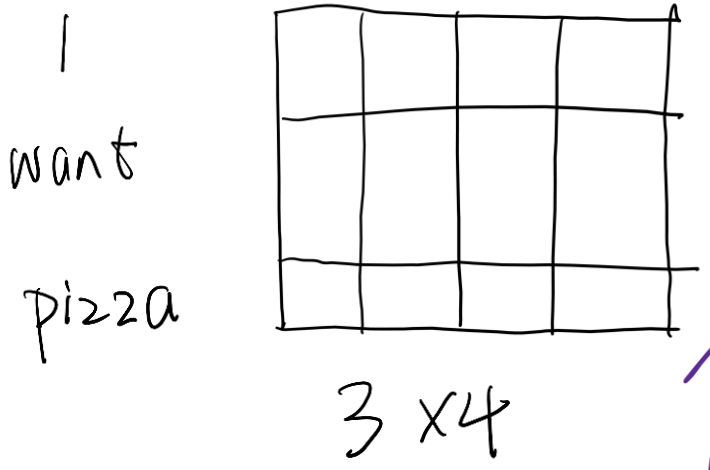


① translation

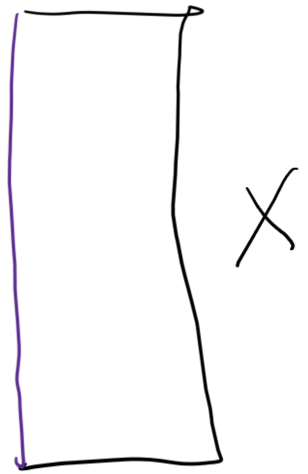
② there is ground truth labels to compare

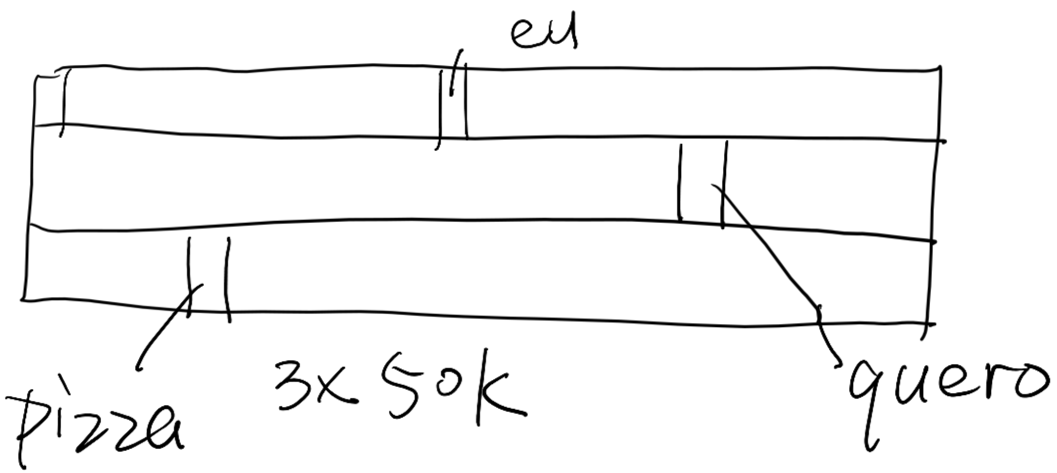
GPT1 &2

Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.

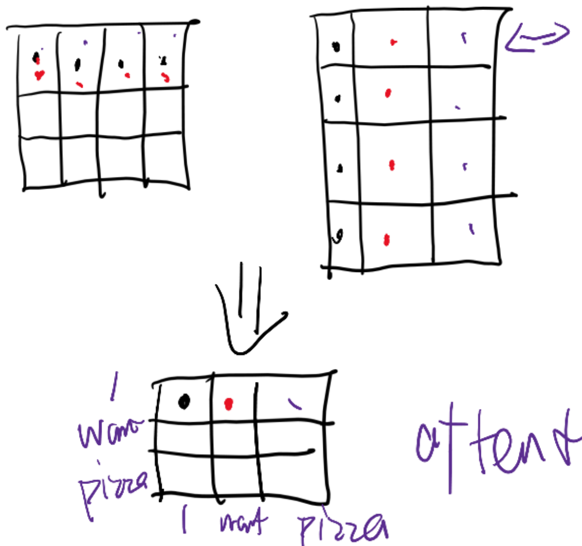


K Q V





multihead head attention



attention

$$\boxed{\text{softmax}(K\,Q^T)\,V}$$

$3\times4 \quad 4\times3 \quad\quad 3\times4$

3×3

$3\times4 \quad \times n$

GPT 3

- Parameters 175 B
- Dataset 45T
- 96 attention heads
- 2048 token size
- Learn from their chief scientist: https://www.youtube.com/watch?v=kCc8FmEb1nY