

Data Mining



Artificial Neural Network and Applications

Group 4

Members

- Thái Phúc Hiệp - 1812227
- Nguyễn Thị Hương - 1970207
- Trần Xuân Hoàng - 1812302
- Nguyễn Xuân Vĩnh Hưng - 1970589



Outline

- Giới thiệu về ANN
- Mô hình Perceptron
- Mạng Neuron
- Hàm Activation
- Multi-class classification
- Cost Function và Gradient Descent
- Backpropagation
- Application
- Demo

Artificial Neural Network

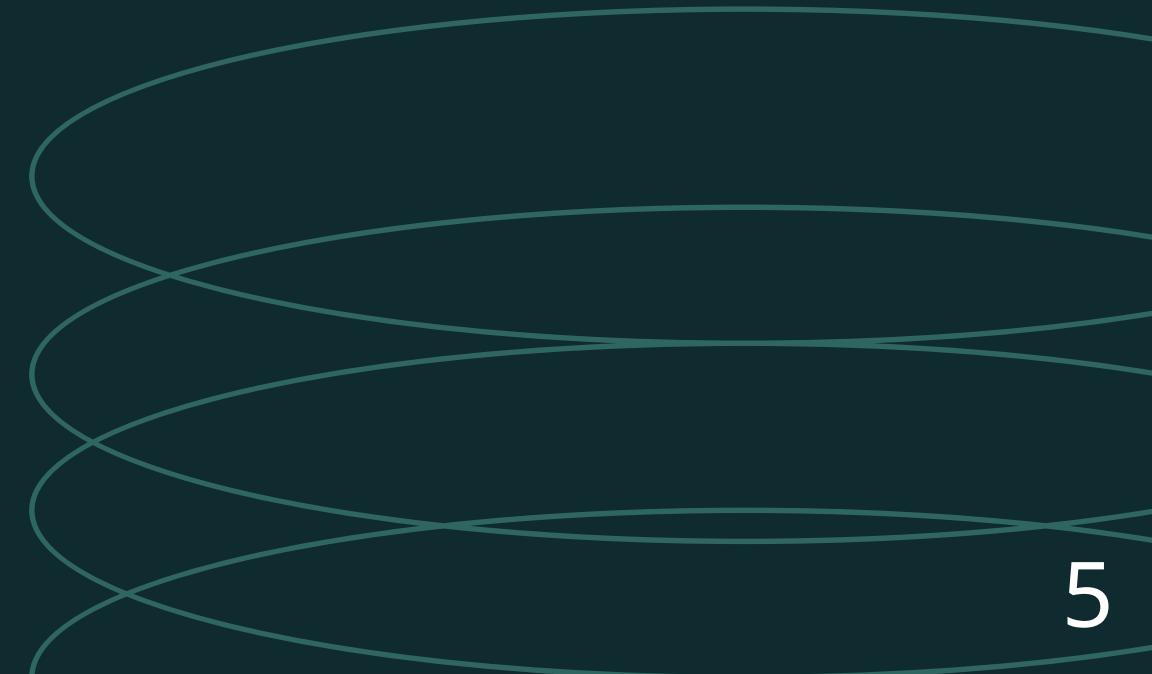
Đặt vấn đề : phân biệt các hình sau

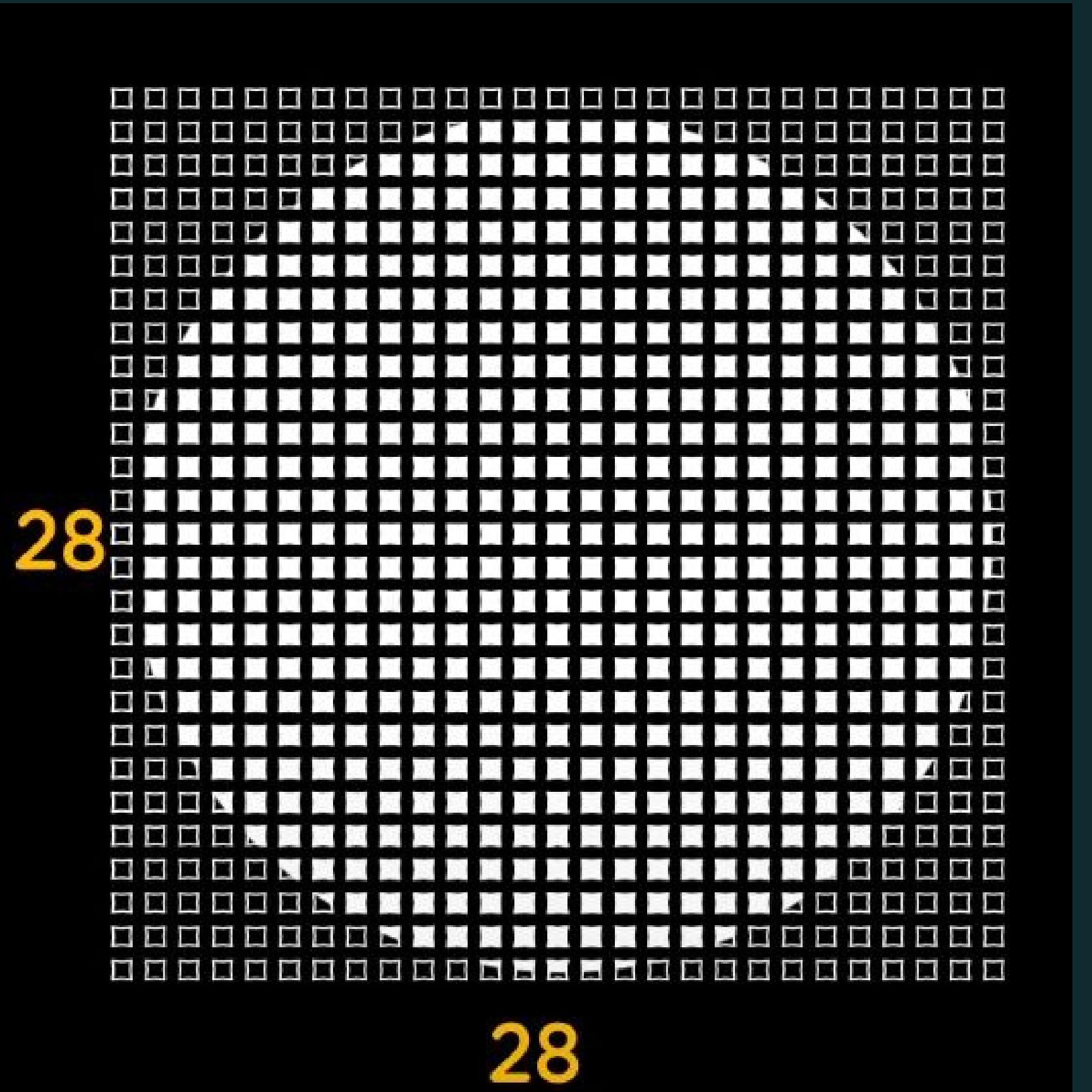


Con người → Dễ dàng phân biệt bằng mắt thường

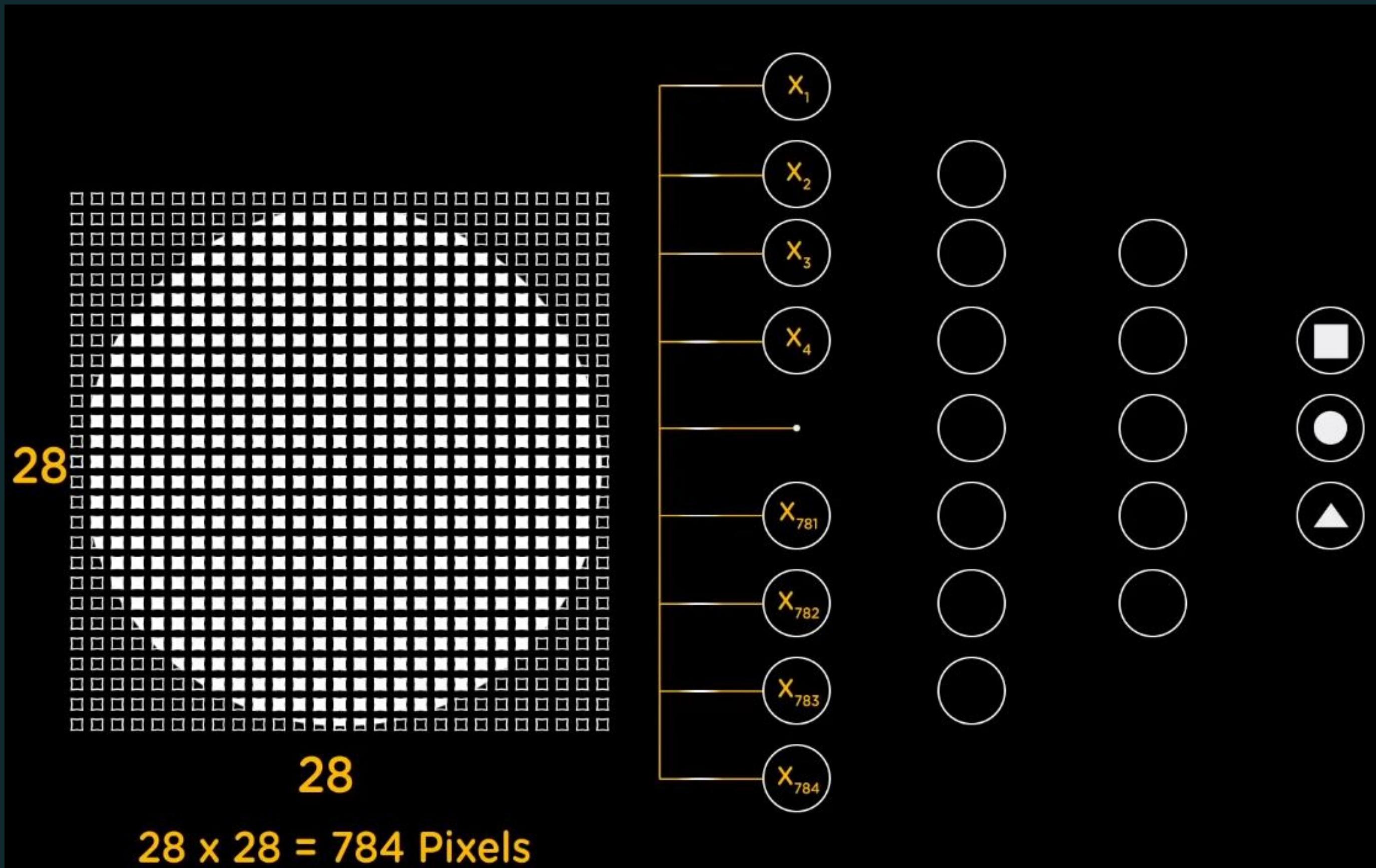
Máy tính → Phải xử lý ảnh

Neural Network !!!

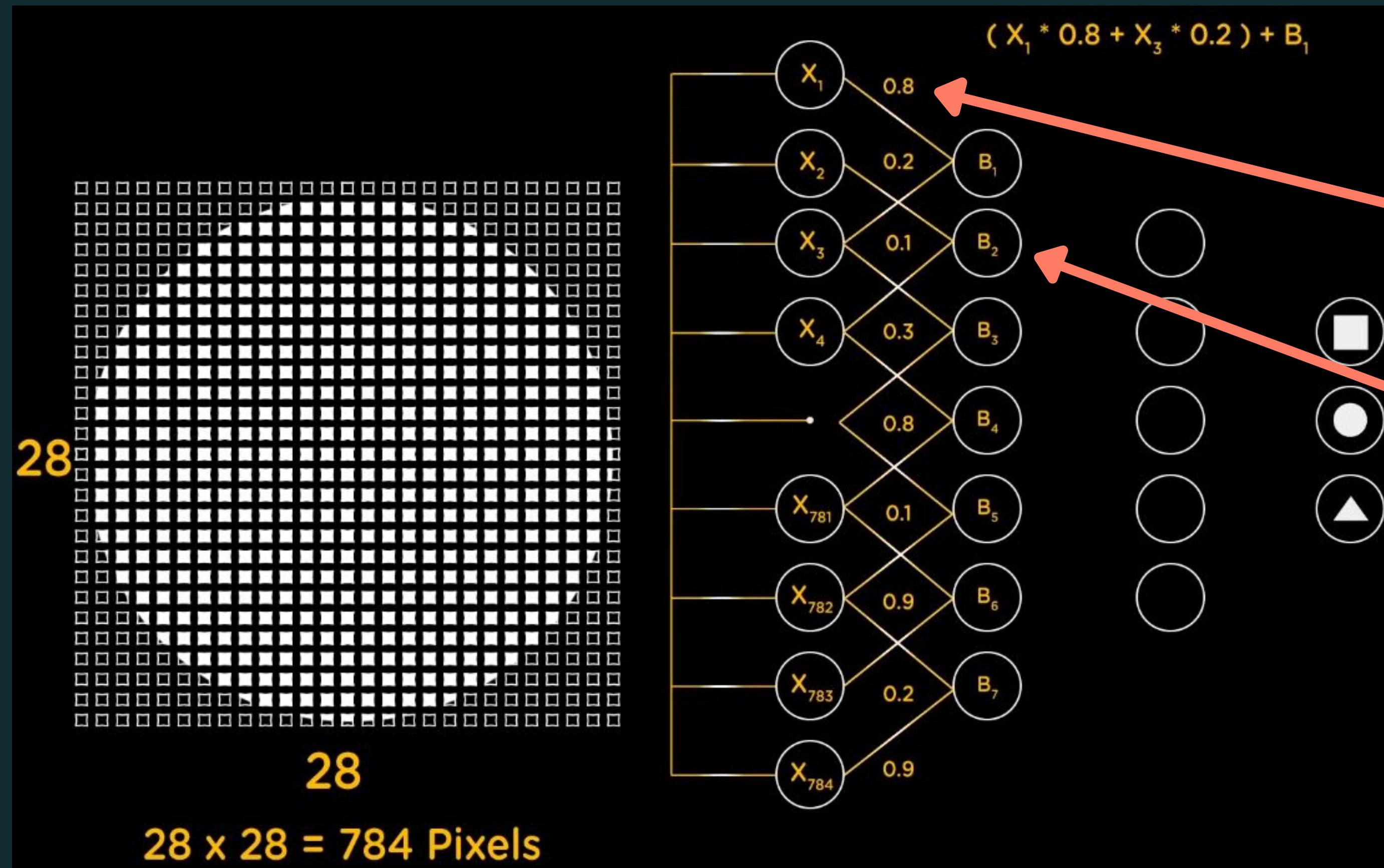




<https://youtu.be/bfmFfD2RIcg>

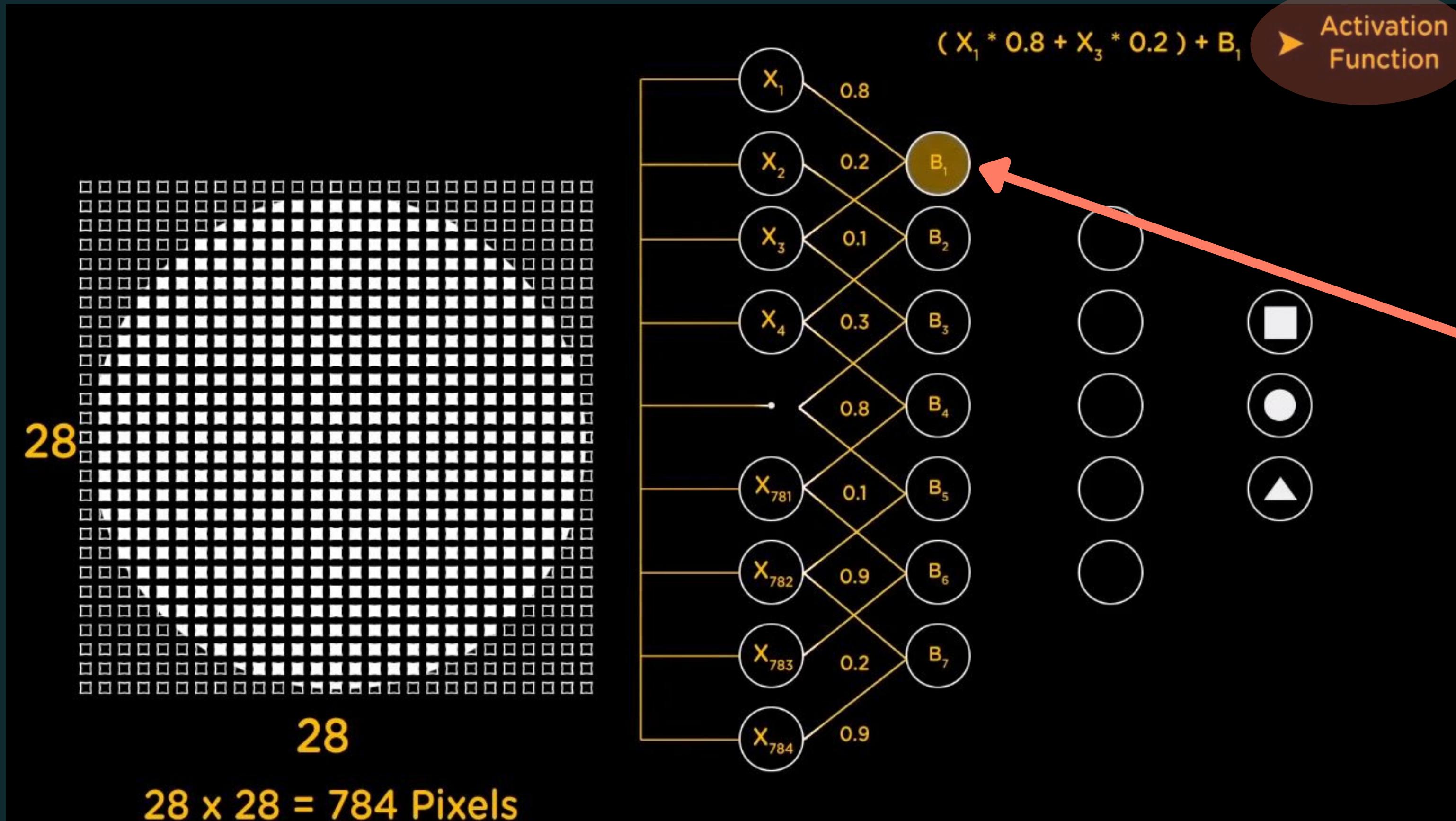


<https://youtu.be/bfmFfD2RIcg>



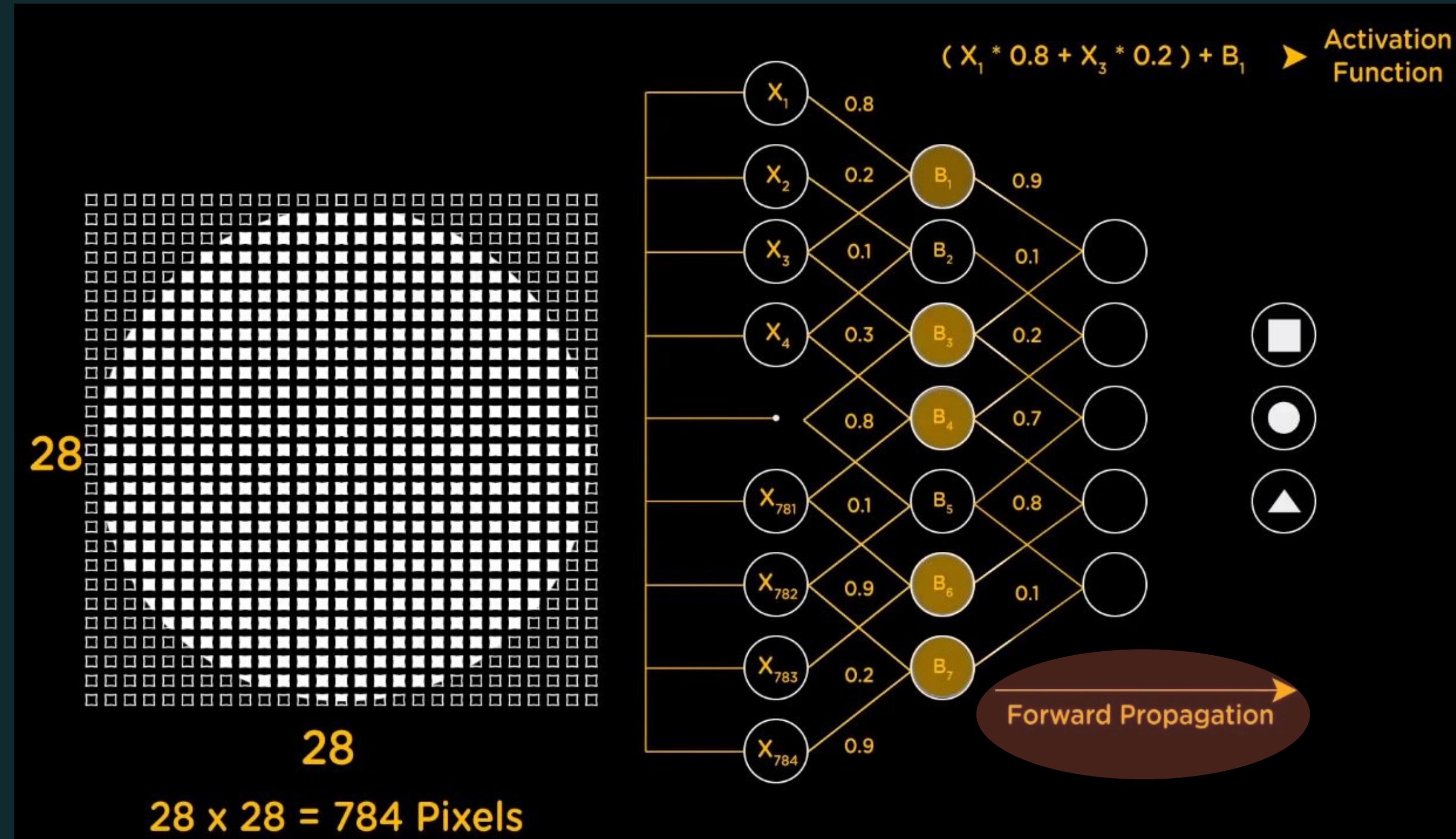
<https://youtu.be/bfmFfD2RIcg>

active
node

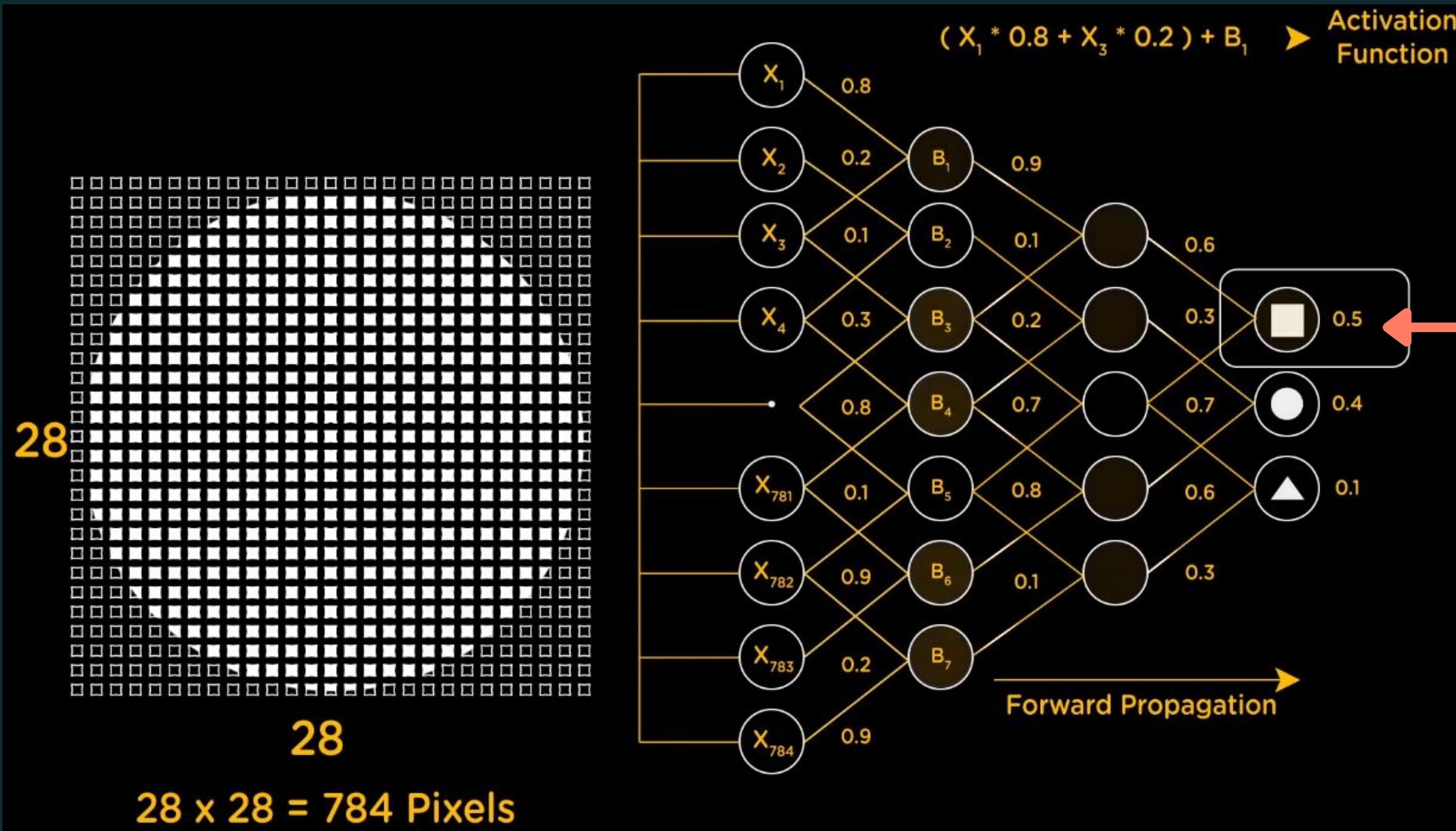


<https://youtu.be/bfmFfD2RIcg>

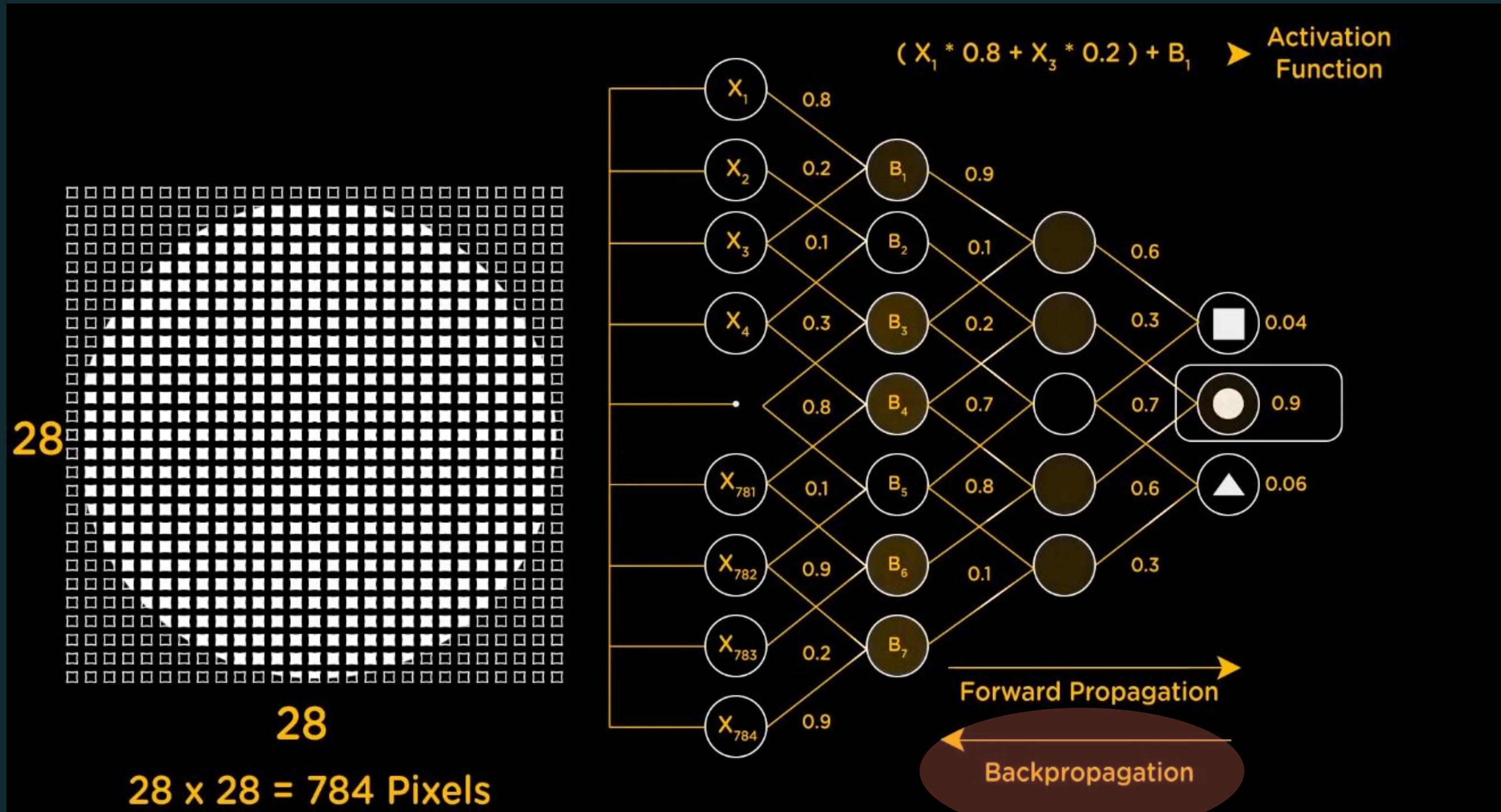
active node sẽ truyền data cho layer tiếp theo



<https://youtu.be/bfmFfD2RIcg>

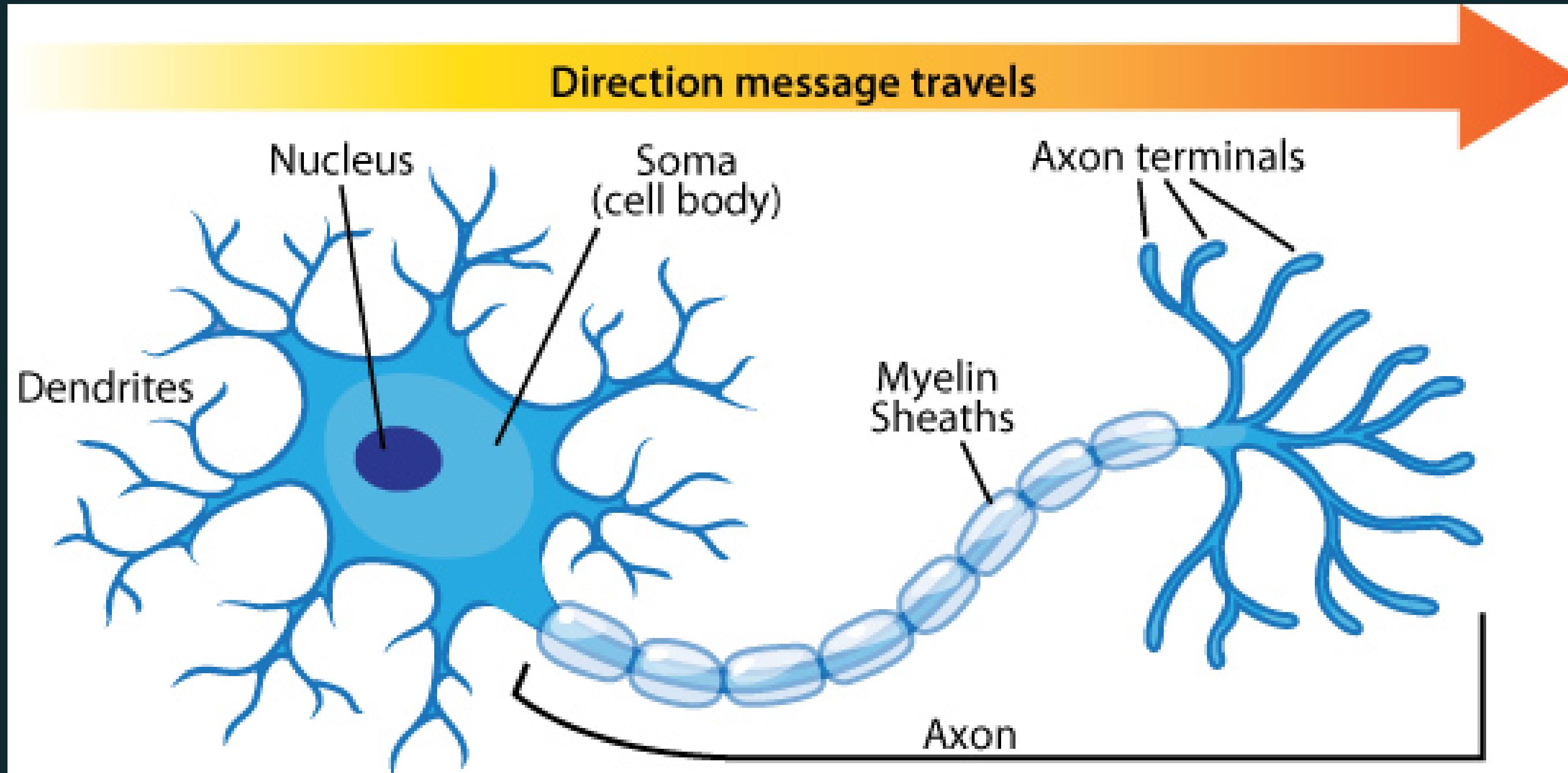


giá trị weight thay đổi trong quá trình train để cho ra kết quả đúng



<https://youtu.be/bfmFfD2RIcg>

Perceptron model

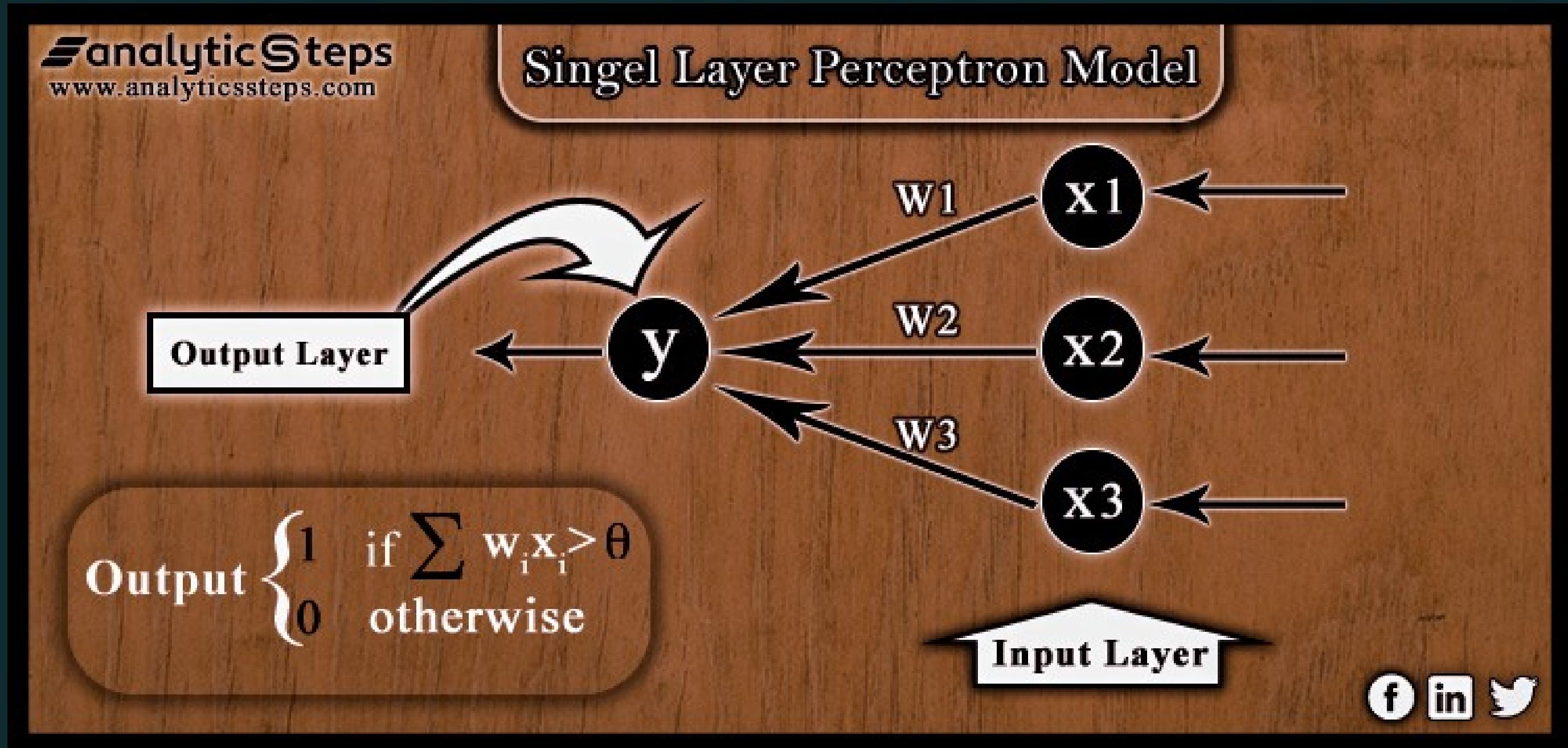


https://askabiologist.asu.edu/sites/default/files/resources/articles/neuron_anatomy.jpg

Perceptron model

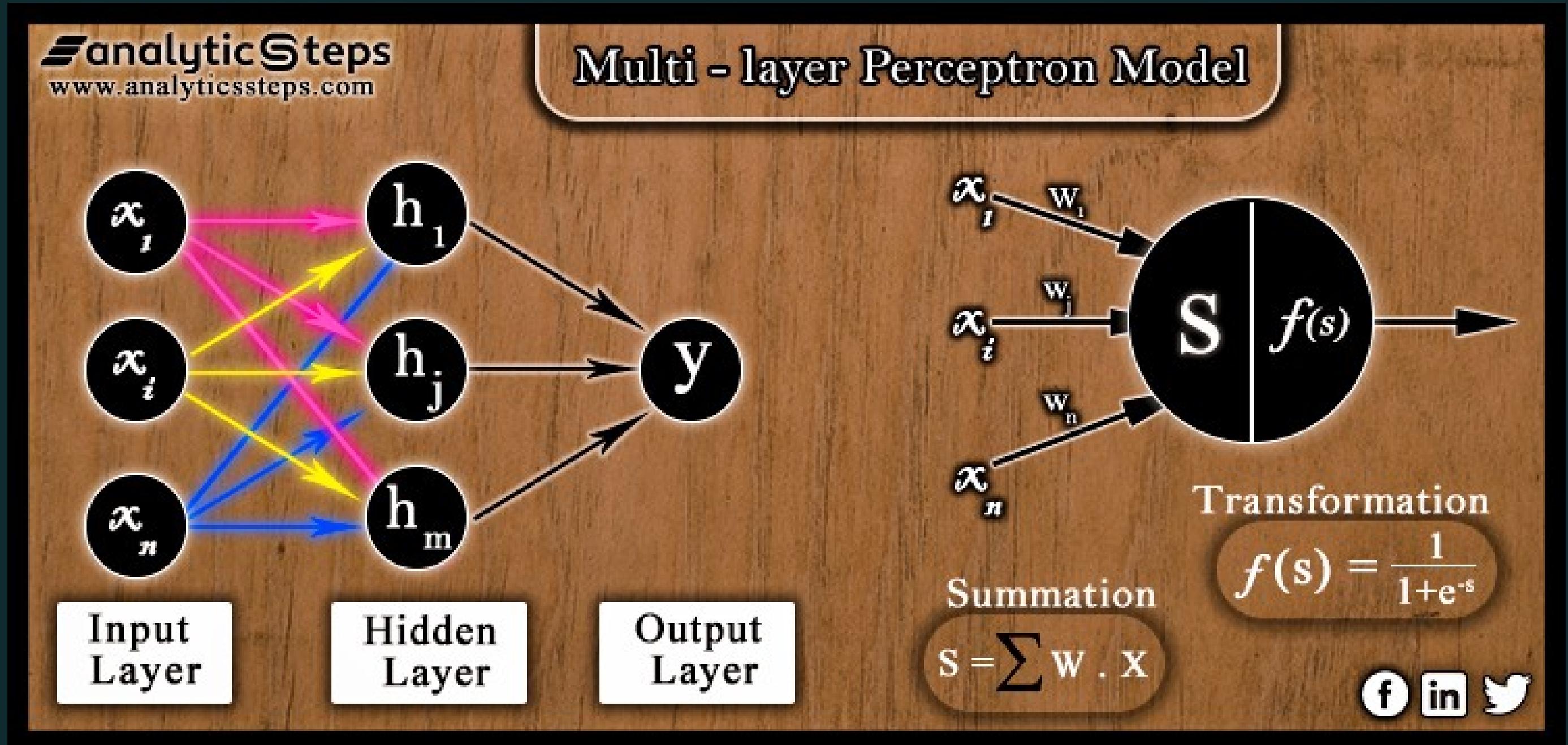
- Perceptron là một mô hình hoạt động giống như một neuron sinh học
- Được phát triển lần đầu năm 1957 tại Cornell Aeronautical Laboratory, Hoa Kỳ
- Bao gồm 2 loại : single-layered perceptron model và multi-layer perceptron model

Single-layered Perceptron model



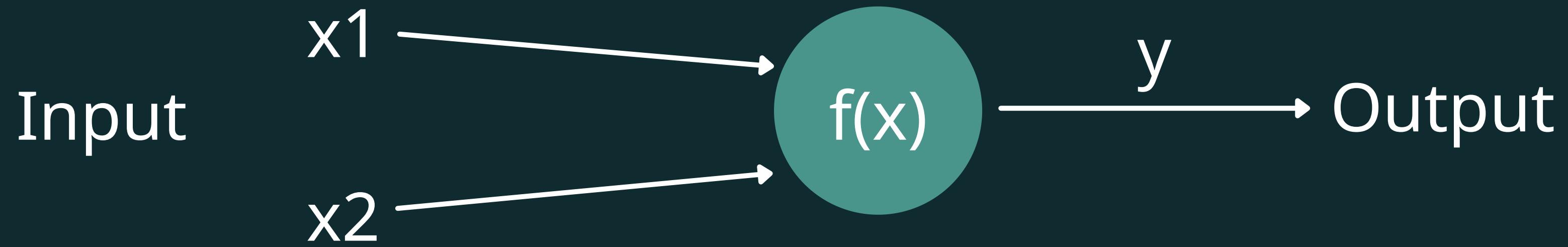
https://miro.medium.com/max/2400/1*hBWTCEctGUwFhw2uy_vqlQ.jpeg

Multi-layered Perceptron model



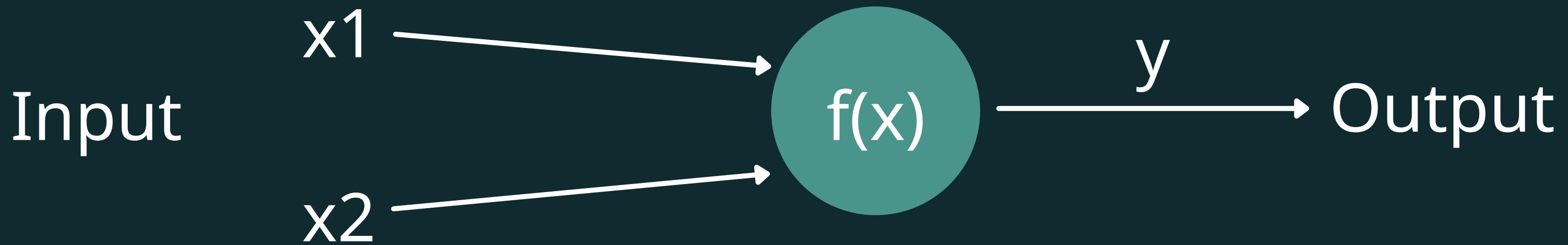
https://miro.medium.com/max/2400/1*zNjqk5e76bLiEF6AgcAVFw.jpeg

Perceptron model



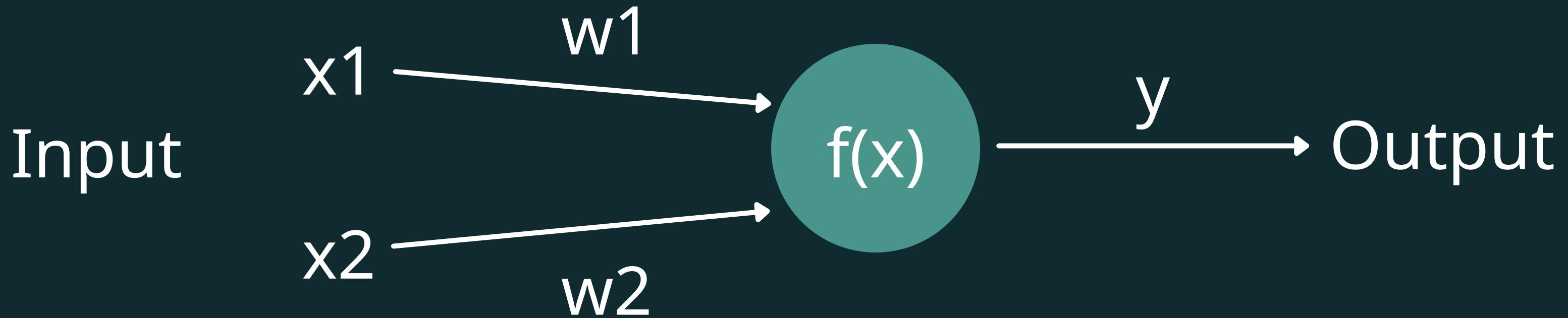
Perceptron model

Nếu $f(x)$ chỉ đơn thuần là phép cộng thì $y = x_1 + x_2$



Perceptron model

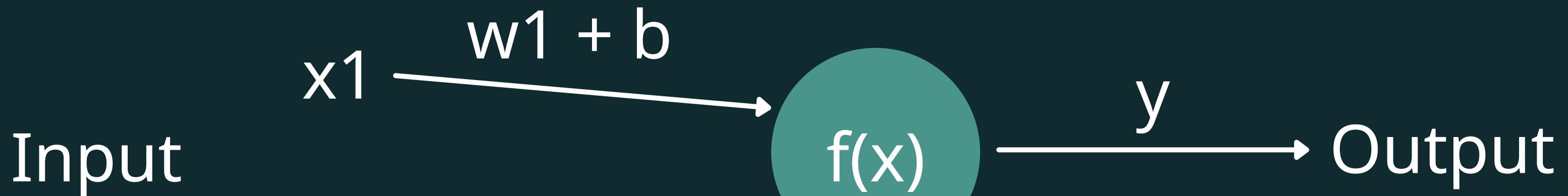
- Trên thực tế, ta cần một tham số để mô hình này có thể "học" được
- Ta cần thêm một giá trị gọi là "weight", giá trị này thay đổi sẽ ảnh hưởng đến y



- Lúc này, $y = x_1w_1 + x_2w_2$

Perceptron model

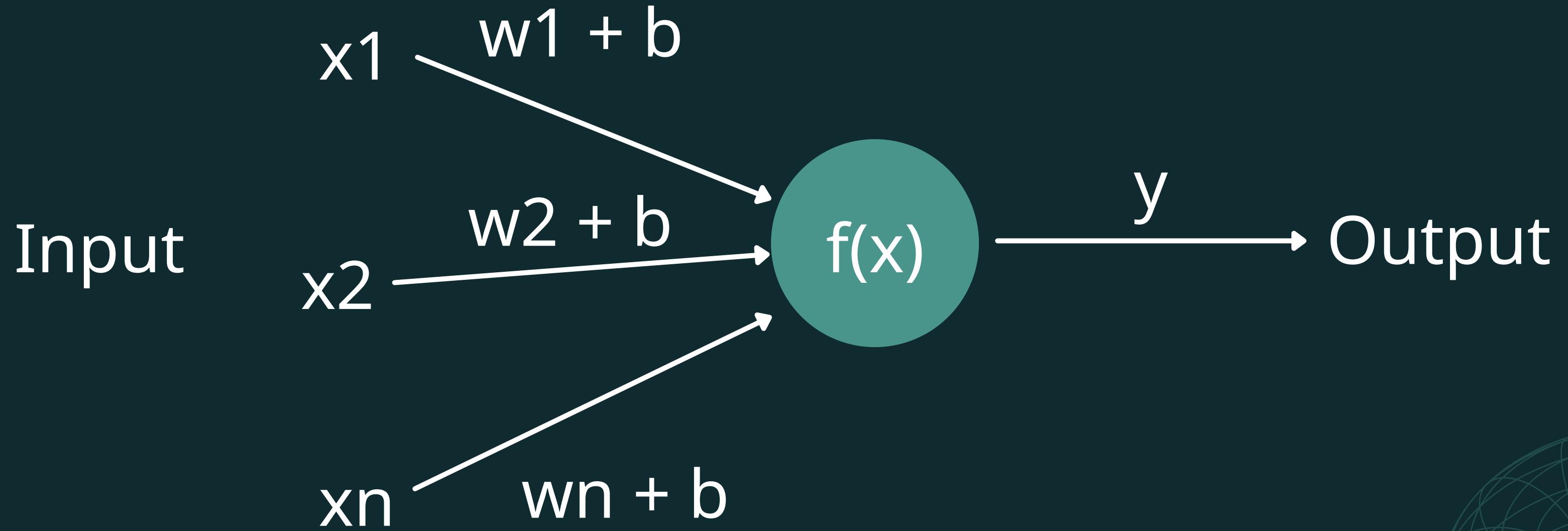
- Nhưng nếu $x = 0$, thì giá trị w sẽ trở nên vô nghĩa
- Ta cần thêm một giá trị gọi là "bias"



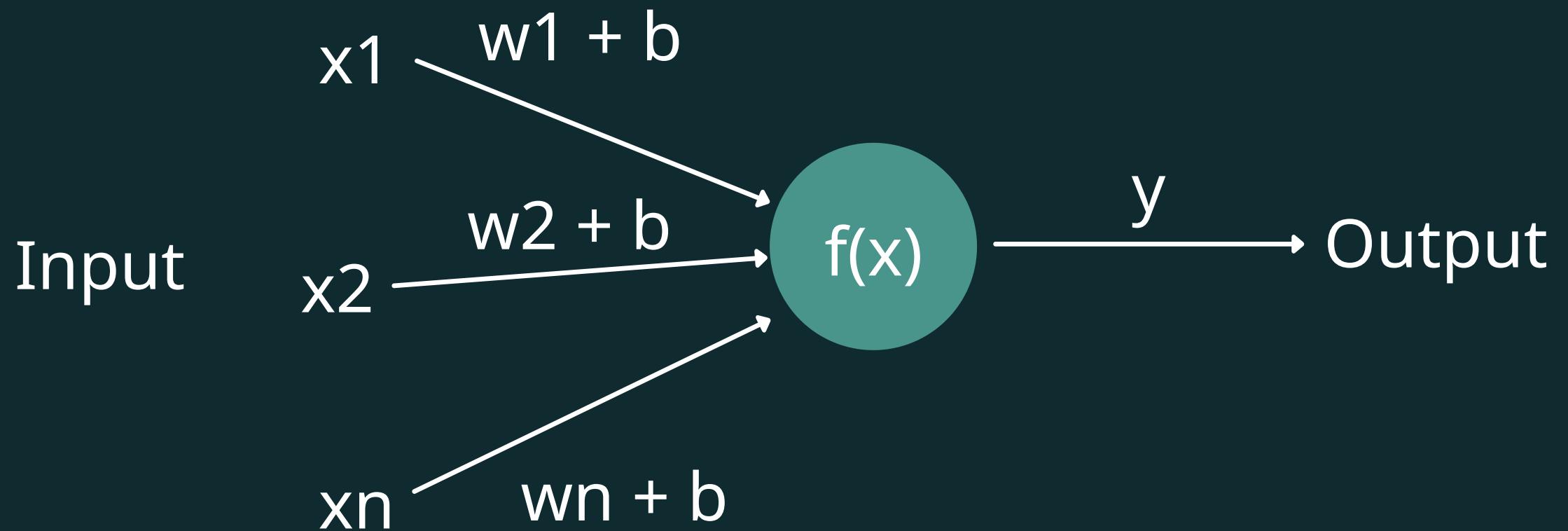
- Lúc này, $y = (x_1w_1 + b) + (x_2w_2 + b)$

Perceptron model

- Tổng quát hóa lên ta sẽ có



Perceptron model

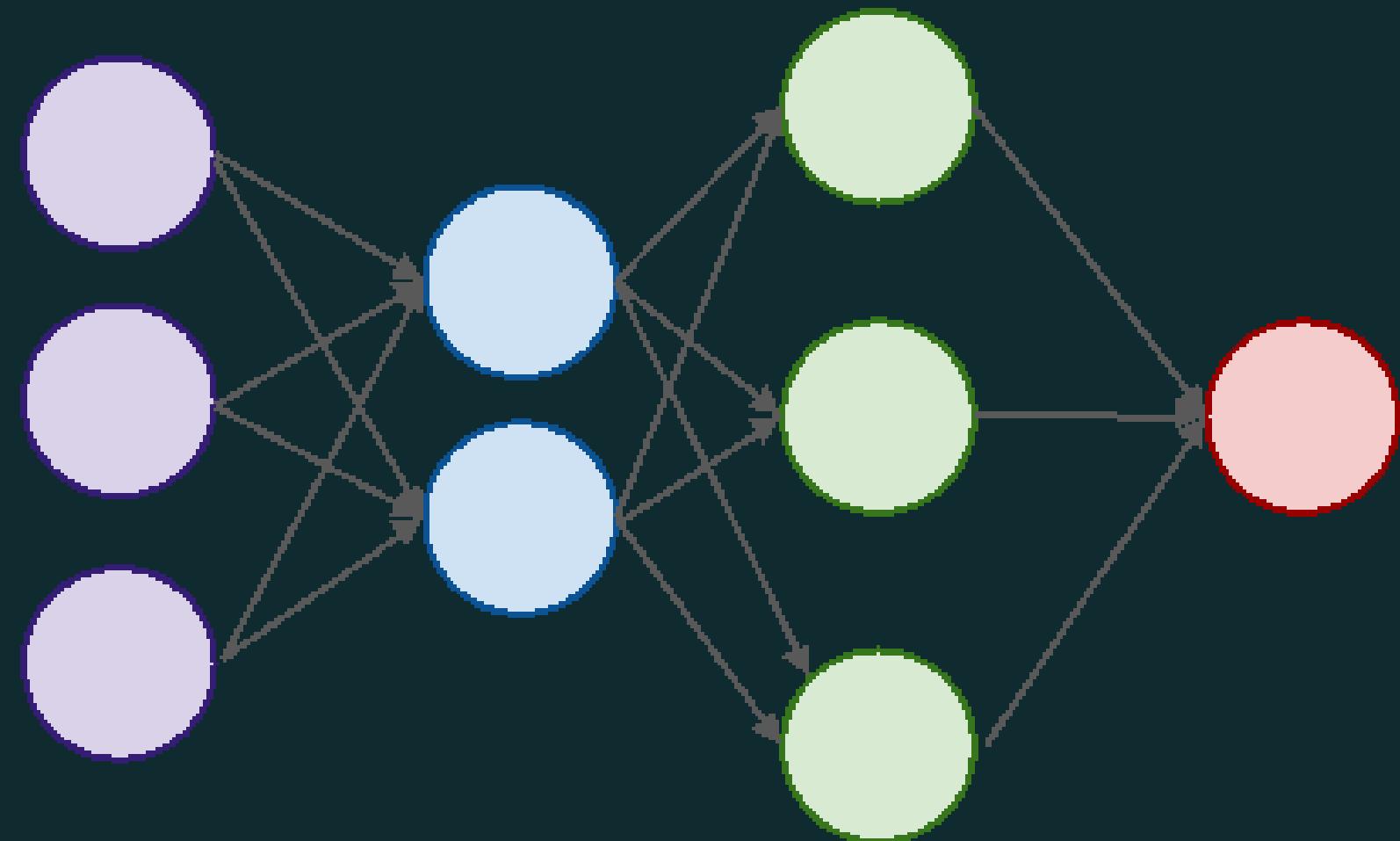


- Được biểu diễn toán học thành

$$\hat{y} = \sum_{i=1}^n x_i w_i + b_i$$

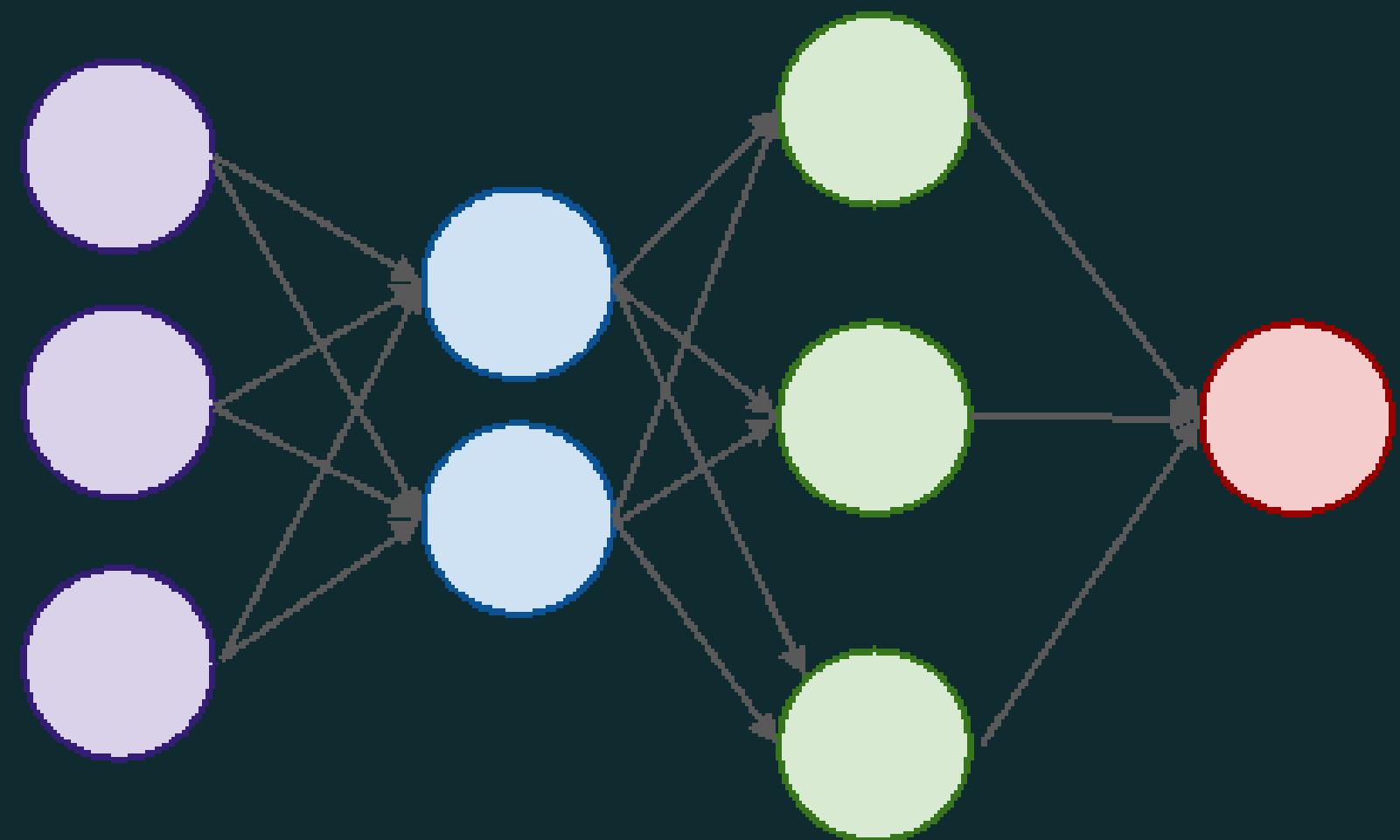
Neural Network

- Được biểu diễn toán học thành
- Một single-layered perceptron sẽ không đủ để có thể học một hệ thống phức tạp, do đó ta sẽ cần tạo ra một multi-layered perceptron



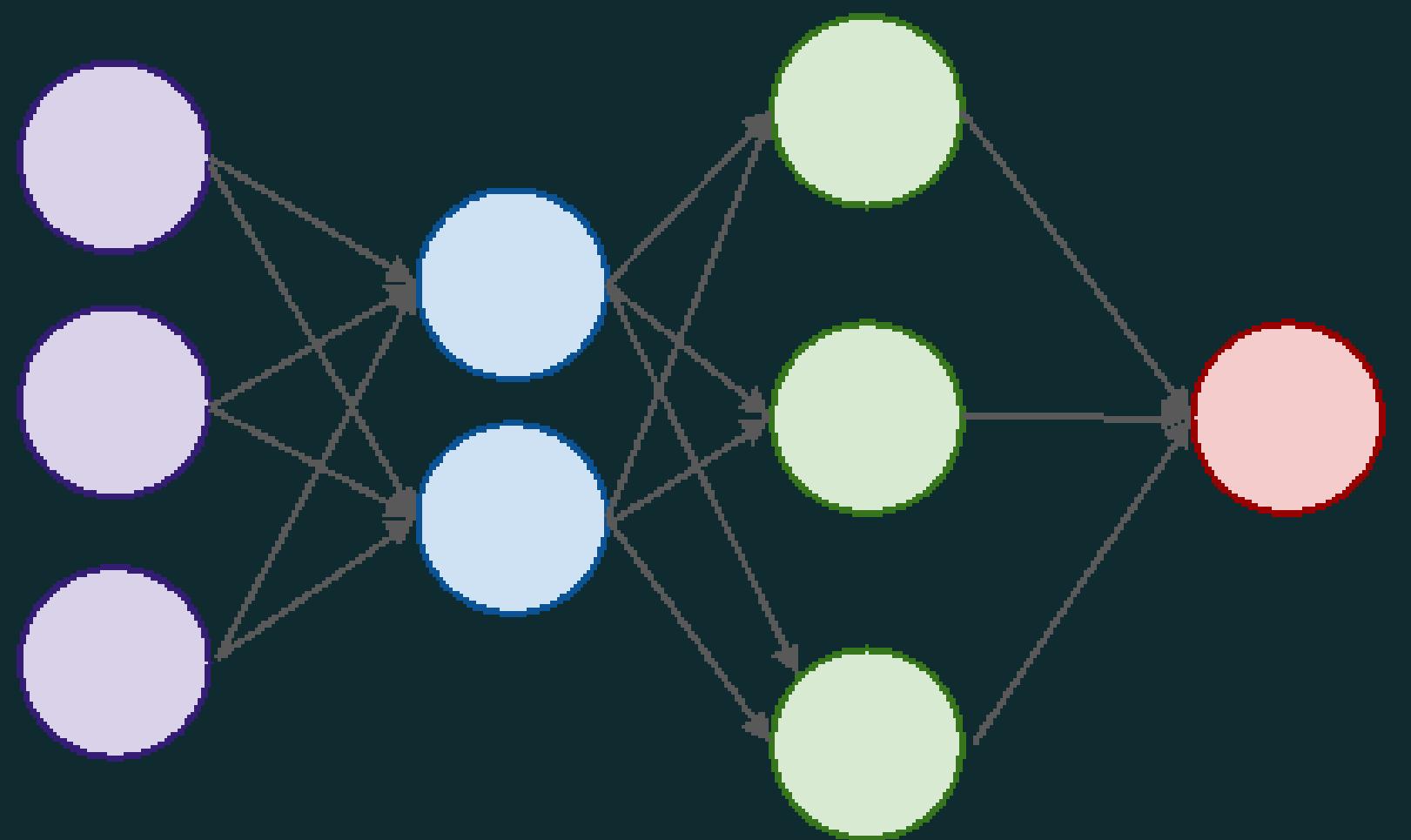
Neural Network

- Để tạo thành một mạng lưới các perceptron, ta sẽ kết nối các lớp (layer) các perceptron lại với nhau



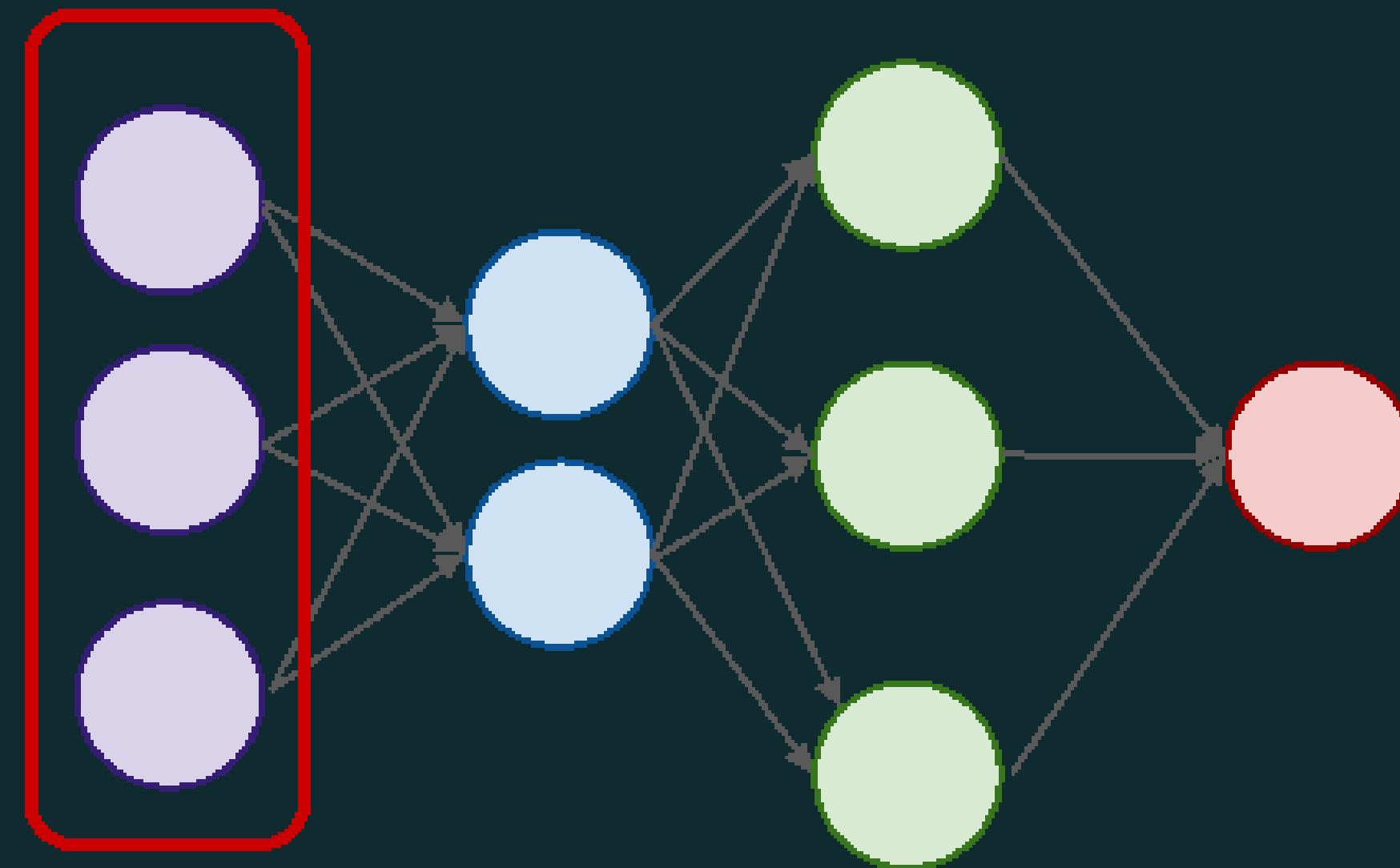
Neural Network

- Output của perceptron này sẽ là input của các perceptron khác
- Điều này sẽ cho phép một mạng neuron có thể học được sự liên quan giữa các đặc tính (feature)



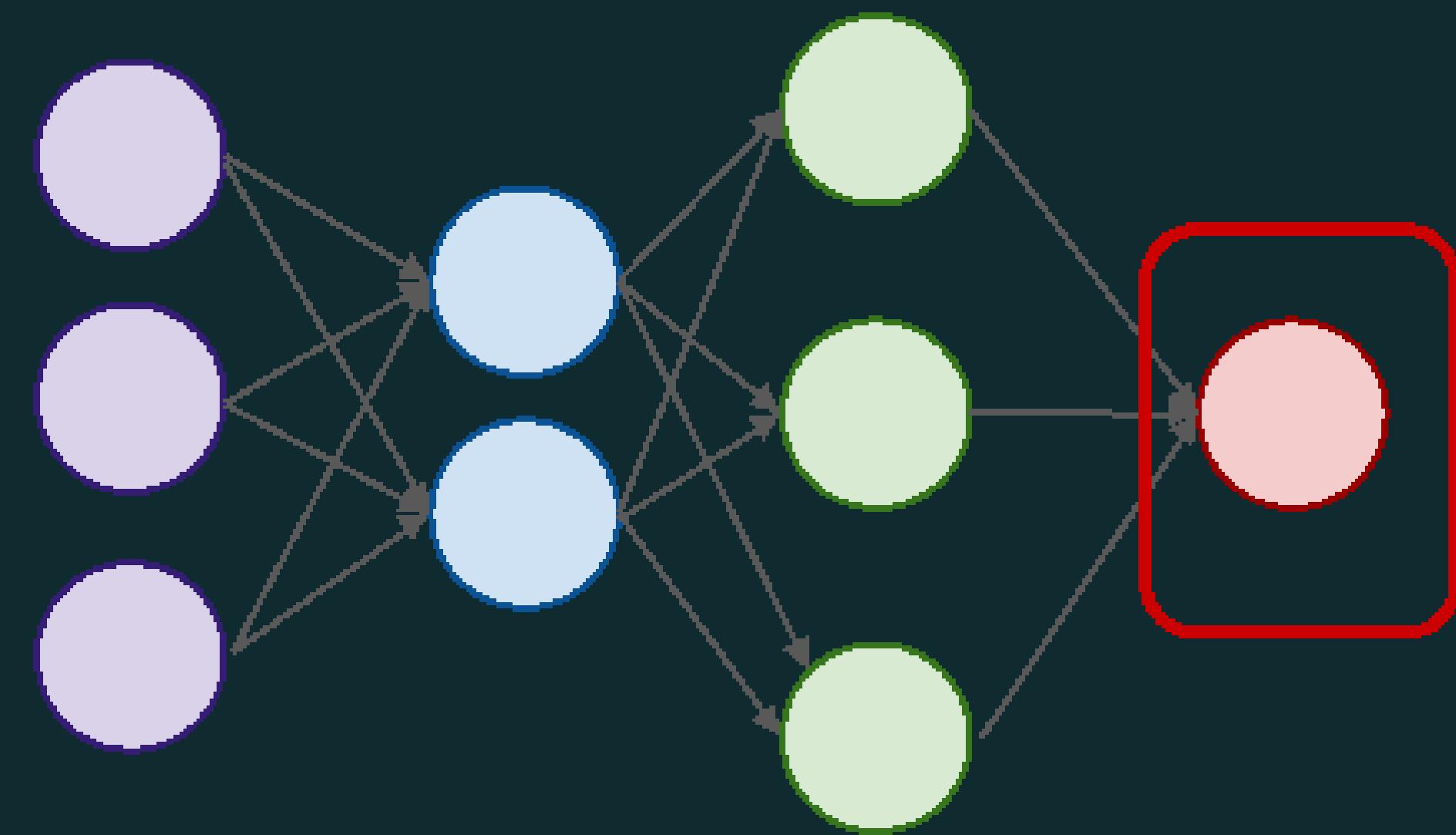
Neural Network

- Lớp đầu tiên được gọi là Input layer



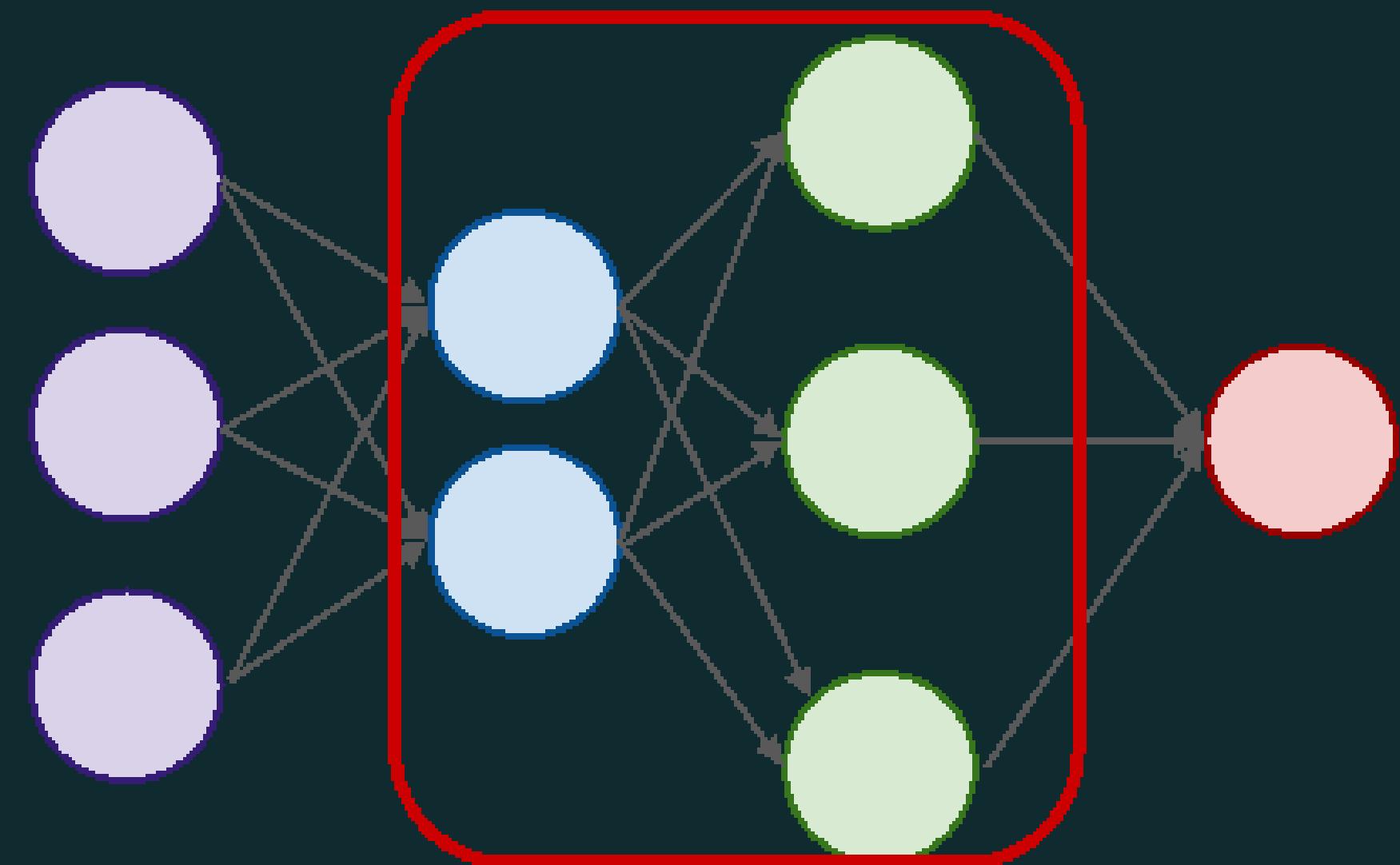
Neural Network

- Lớp cuối cùng gọi là Output layer, và có thể chứa nhiều hơn 1 neuron



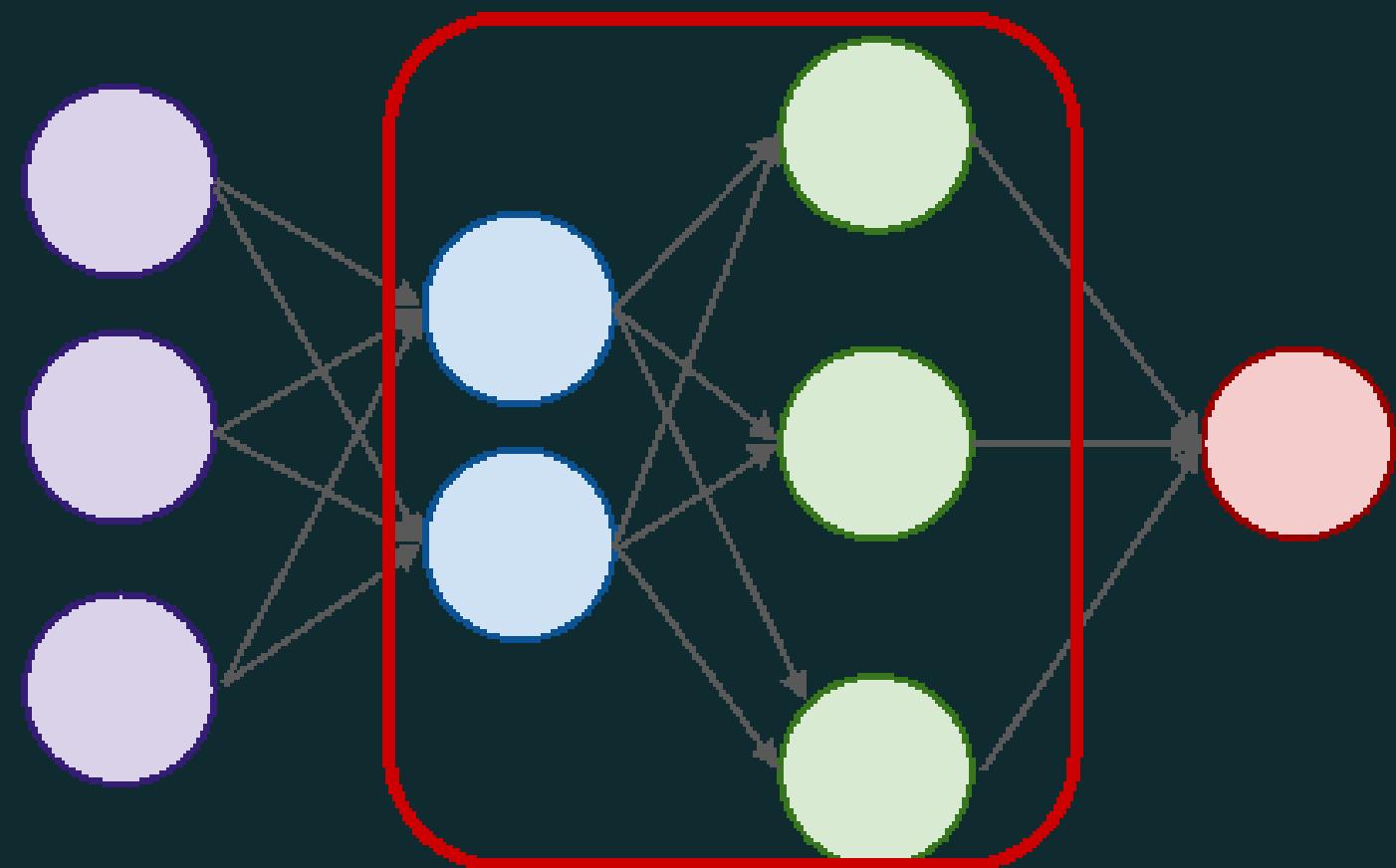
Neural Network

- Các lớp ở giữa được gọi là Hidden Layer



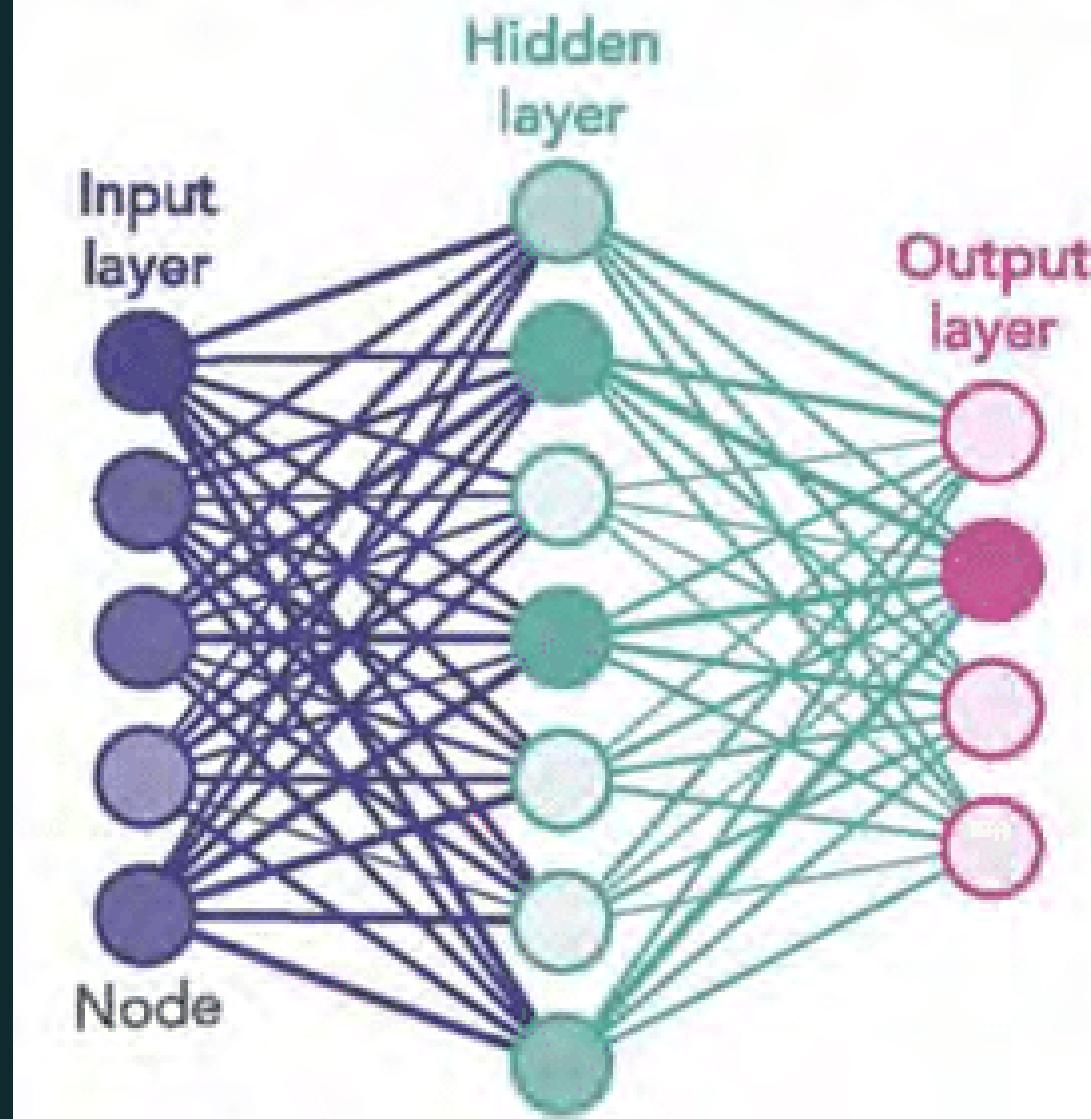
Neural Network

- Các Hidden Layer thường có cấu trúc rất phức tạp, và là nơi sẽ đảm nhận các công việc tính toán
- Một mạng neuron có 2 Hidden Layer trở lên được gọi là Deep Neural Network

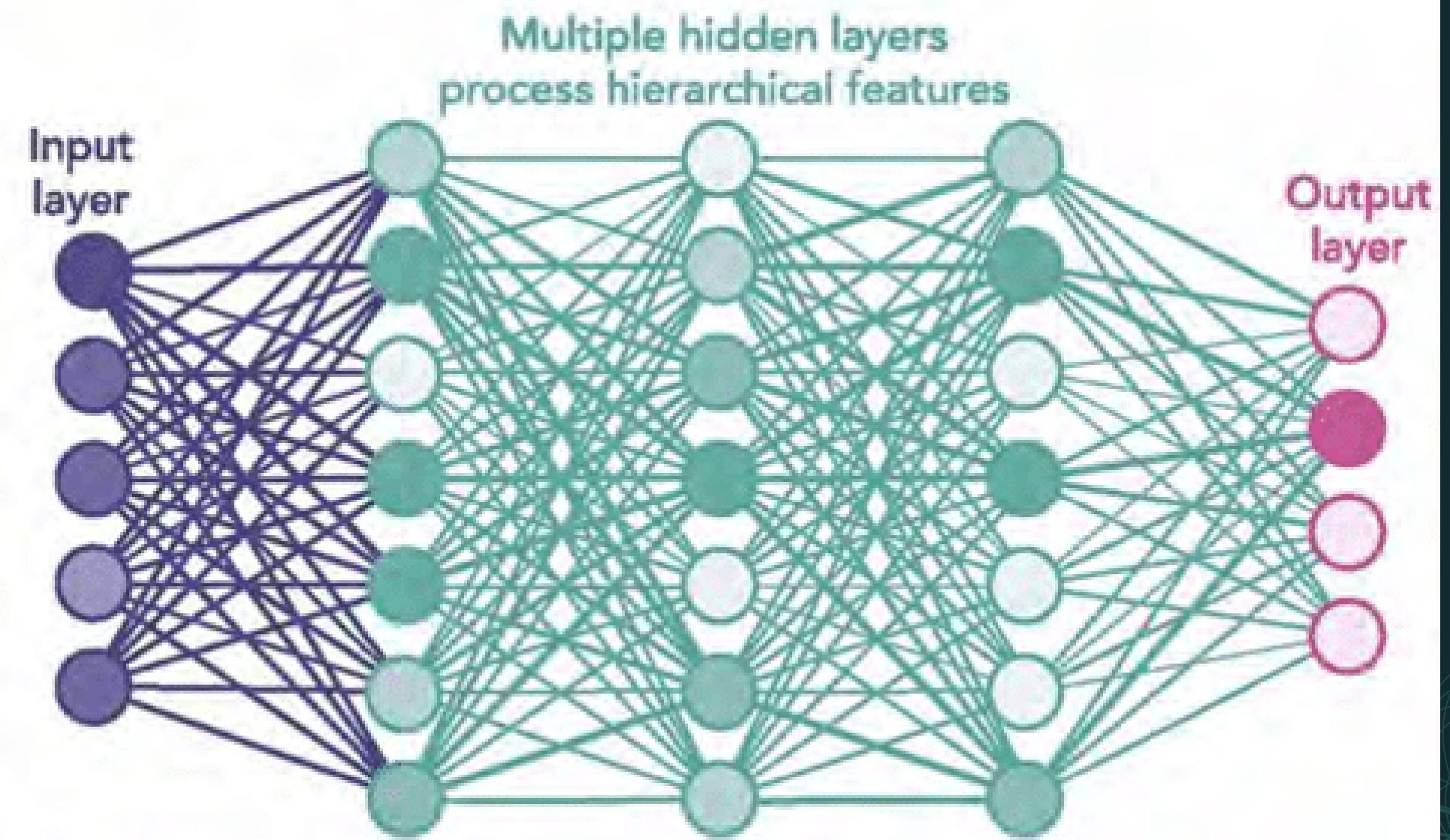


Neural Network

SHALLOW NEURAL NETWORK



DEEP NEURAL NETWORK

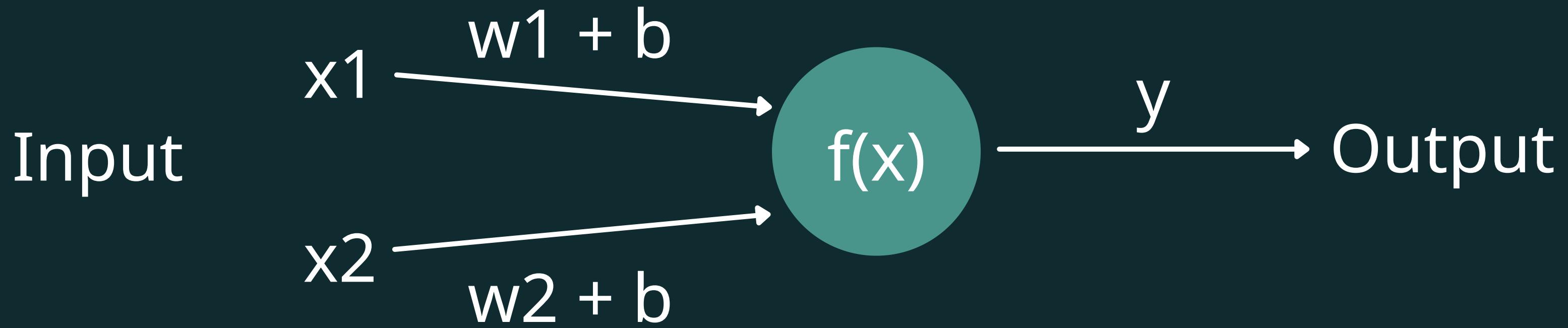


<https://www.researchgate.net/profile/Dongkwon-Han/publication/346219836/figure/fig1/AS:980115399913472@1610689129209/Schematic-of-shallow-neural-network-and-deep-neural-network.ppm>

Activation Function

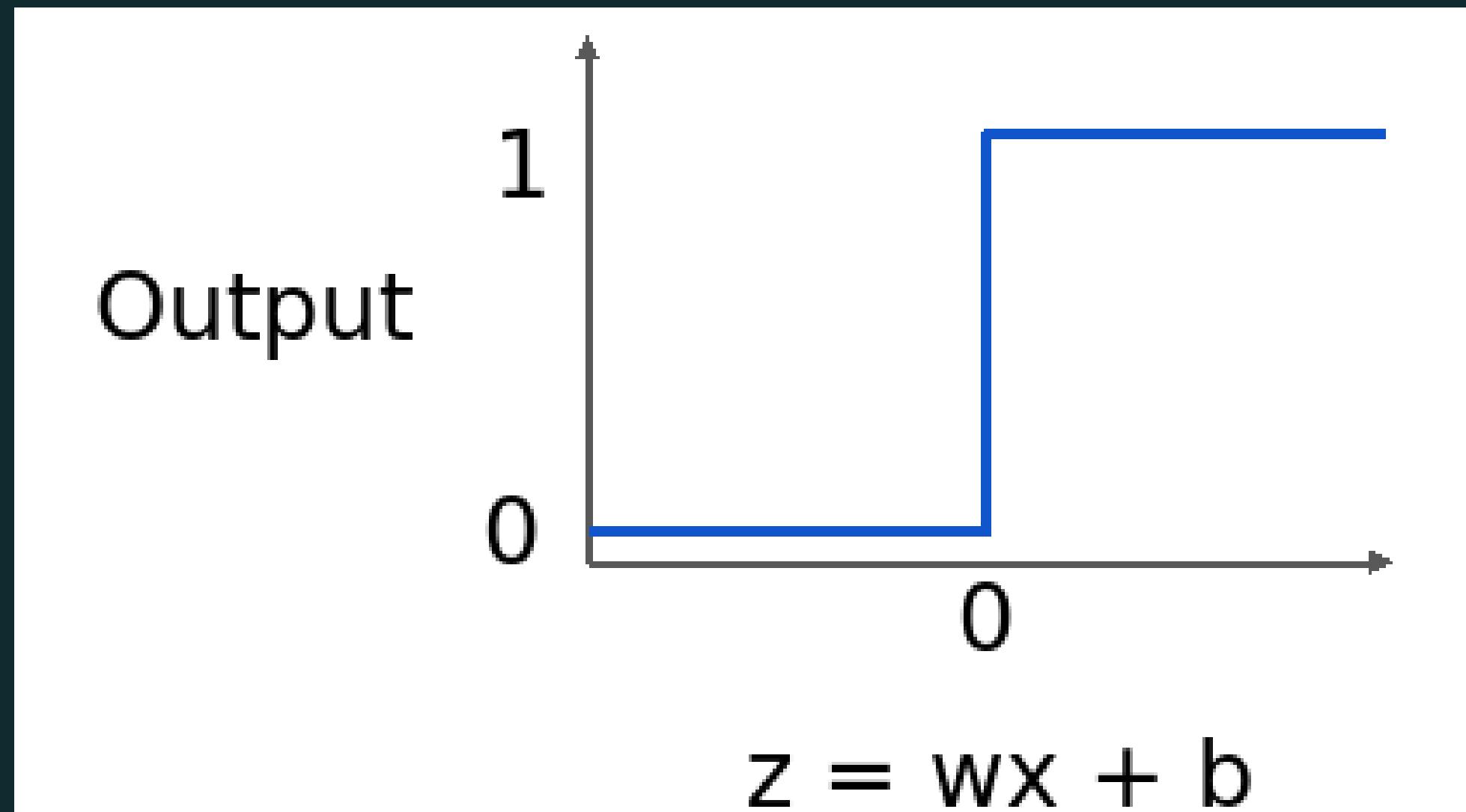
- Nhắc lại : trong một mạng neuron, với input là x ta sẽ có giá trị weight (w) và bias (b)
- Với công thức là : $x * w + b$
- Ta có thể thấy w thể hiện độ quan trọng của input x , còn b là một giá trị offset để $x*w$ có thể đạt tới ngưỡng được active, và lan truyền tới các neuron khác
- Có thể giá trị $x*w + b$ rất lớn nên ta cần một activation function để có thể giới hạn lại giá trị của nó

Activation Function



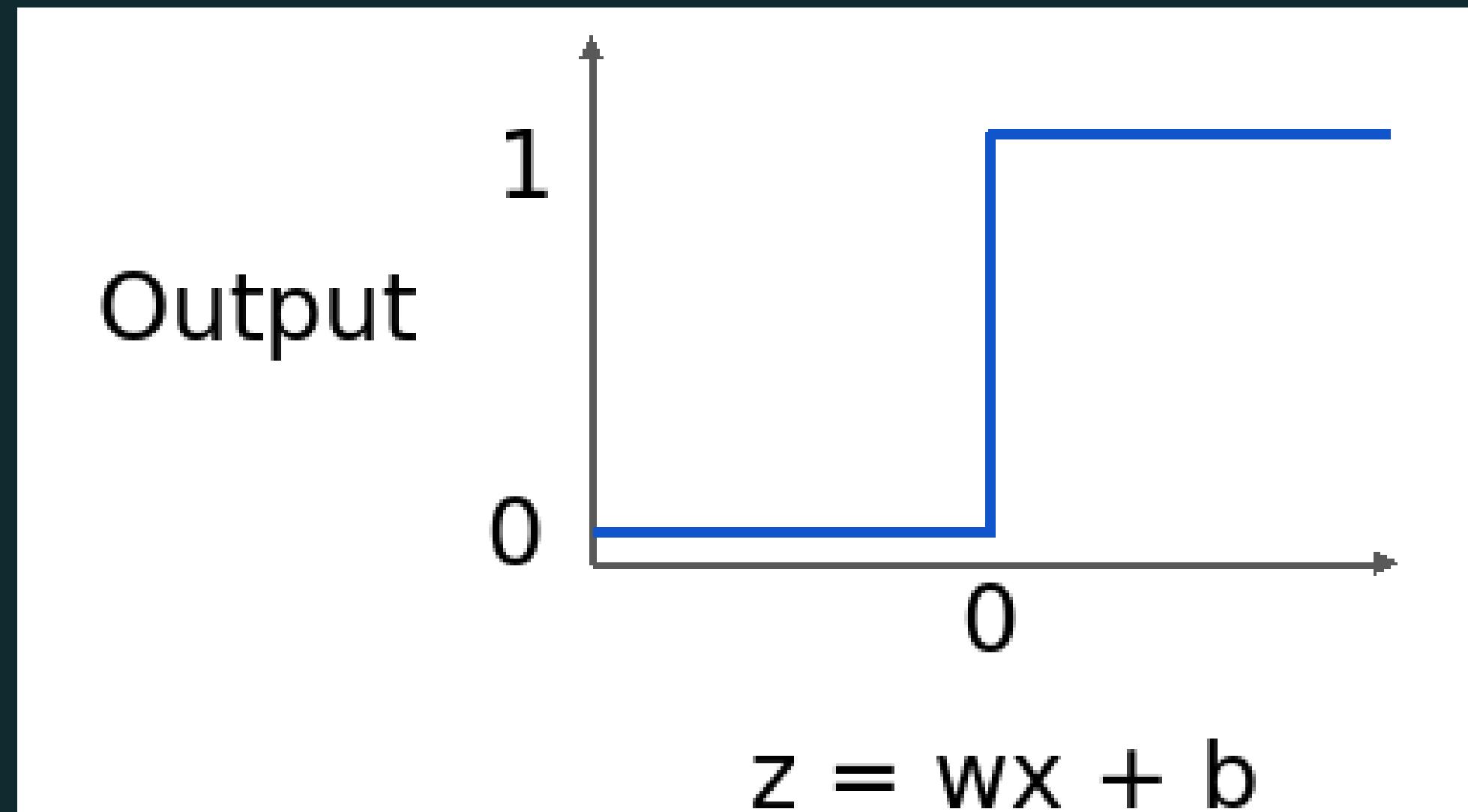
- Giả sử ta có một perceptron như hình, nếu bài toán ở đây là classification thì ta muốn output y chỉ nằm trong vùng giá trị từ 0 tới 1

Activation Function



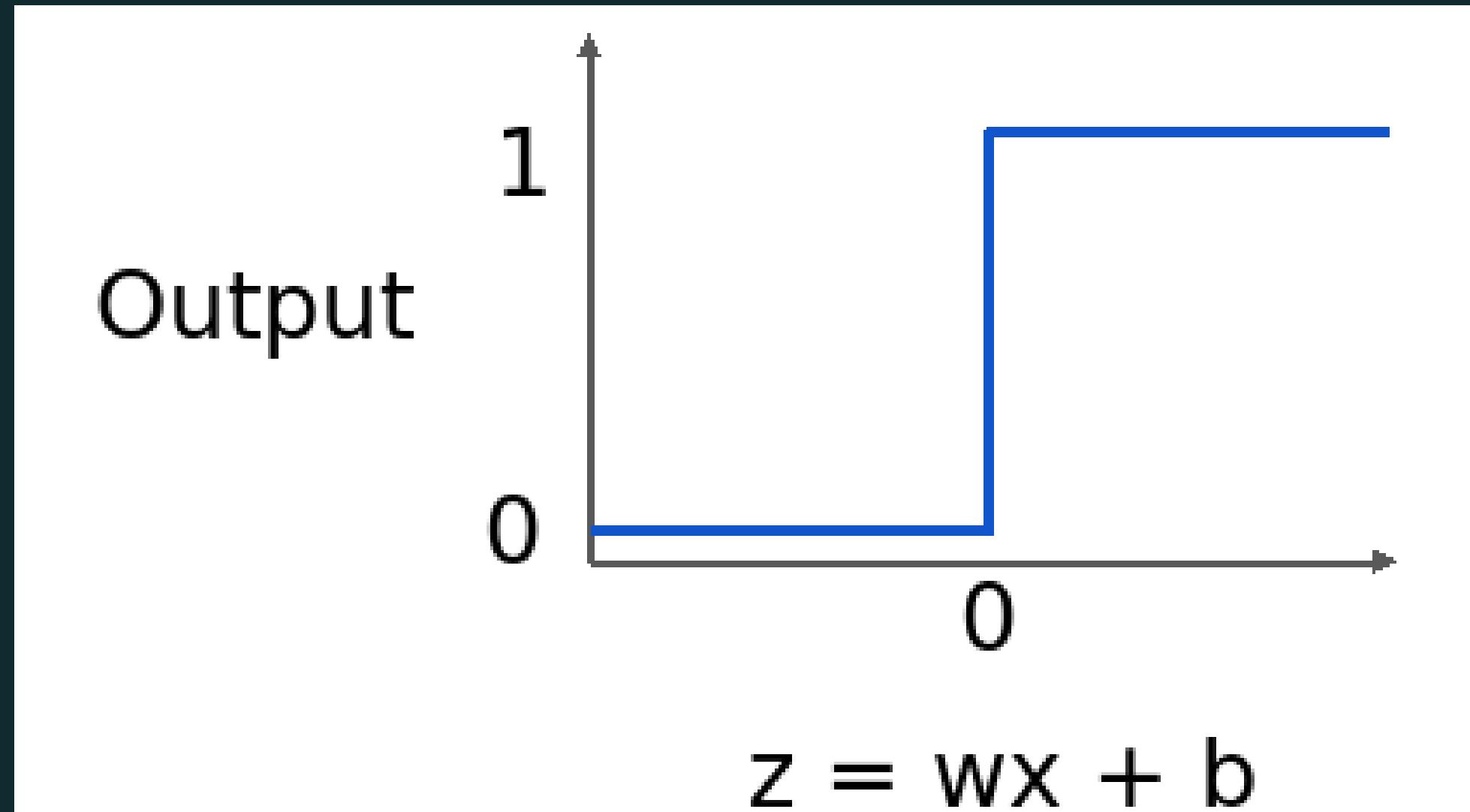
- Một hàm số như thế này có thể đáp ứng được yêu cầu của chúng ta

Activation Function



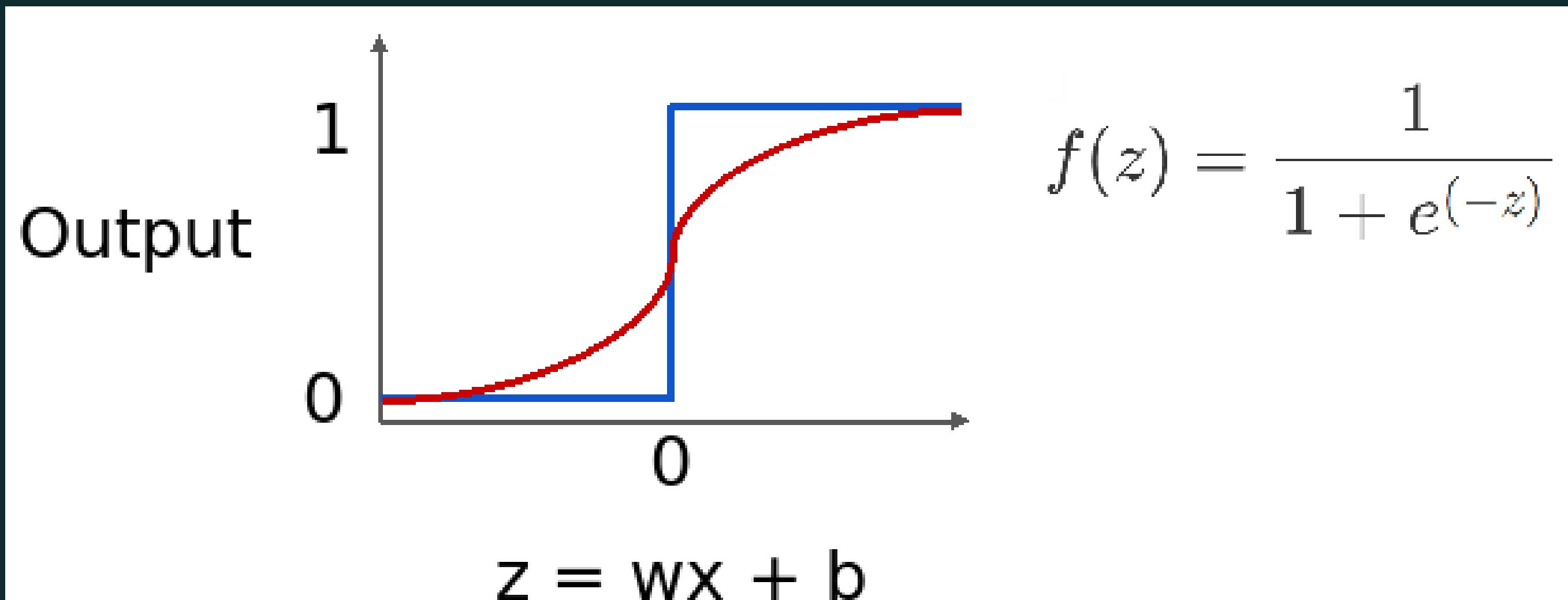
- Cho dù giá trị input có ra sao đi nữa thì giá trị output vẫn chỉ là 0 hoặc 1

Activation Function



- Nhưng điều này thường không tốt vì nó sẽ không phản ánh được các thay đổi nhỏ

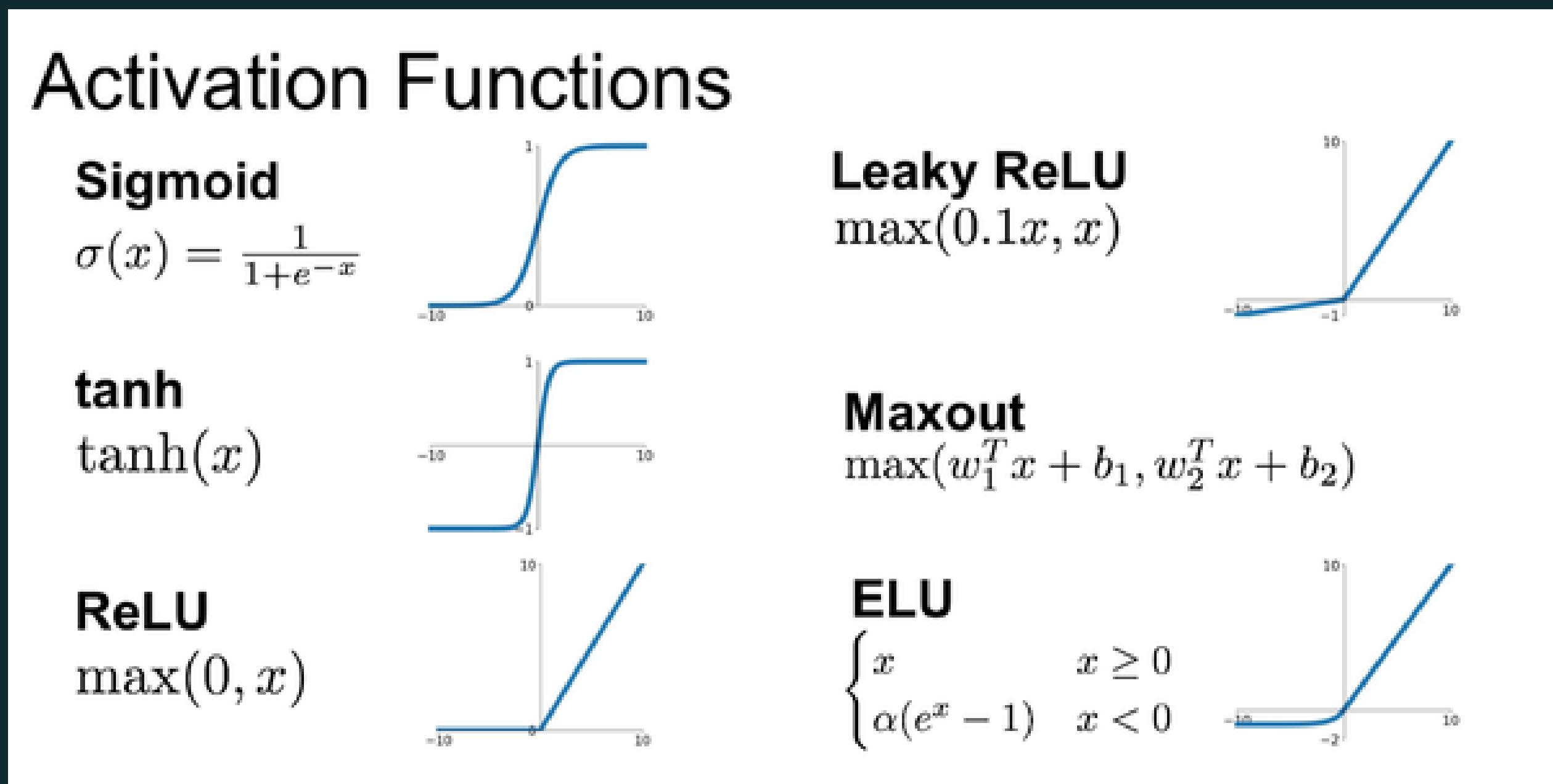
Activation Function



- Do đó, ta cần một hàm số khác tối ưu hơn, và đó chính là hàm sigmoid

Activation Function

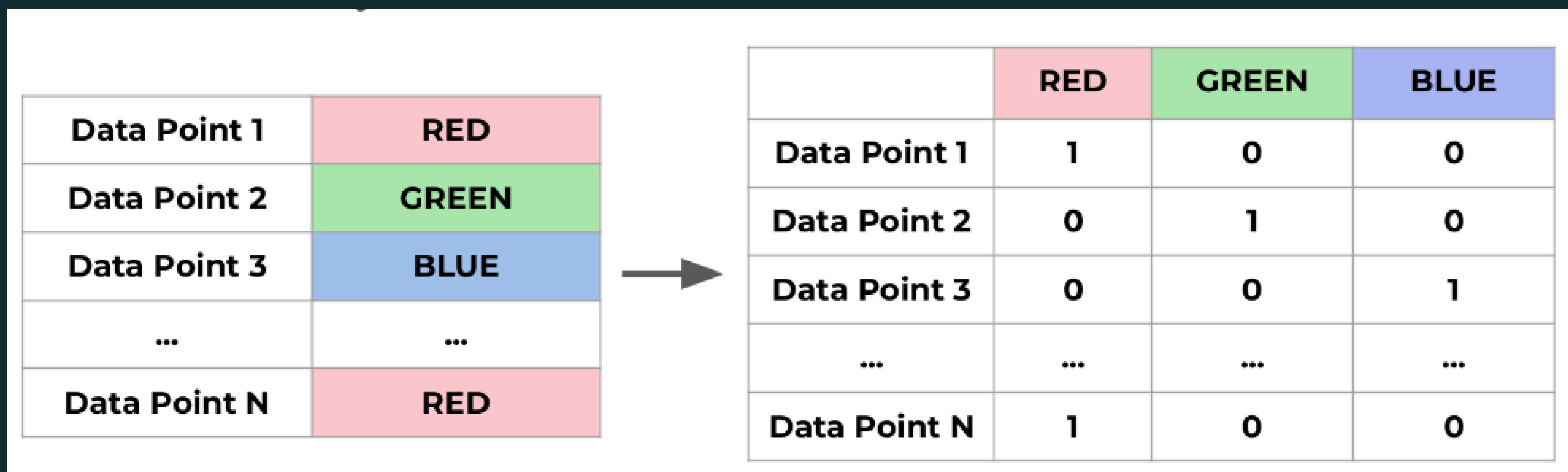
- Một số activation function thường gặp



Multi-class

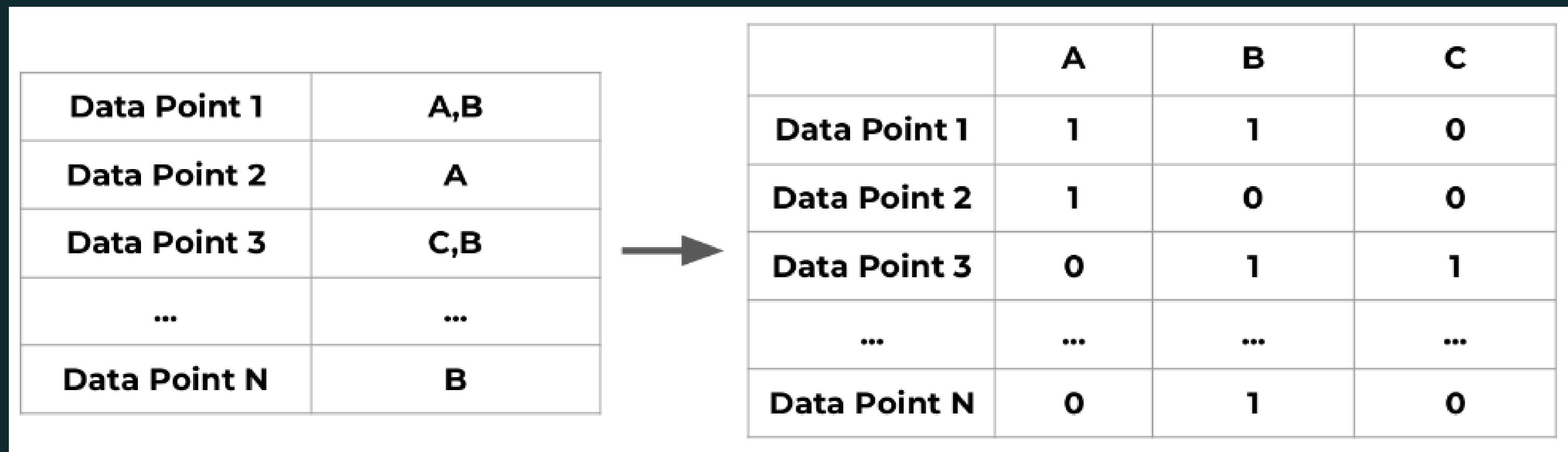
Multi class có thể chia thành 2 trường hợp chính

Mỗi điểm dữ liệu chỉ thuộc một class

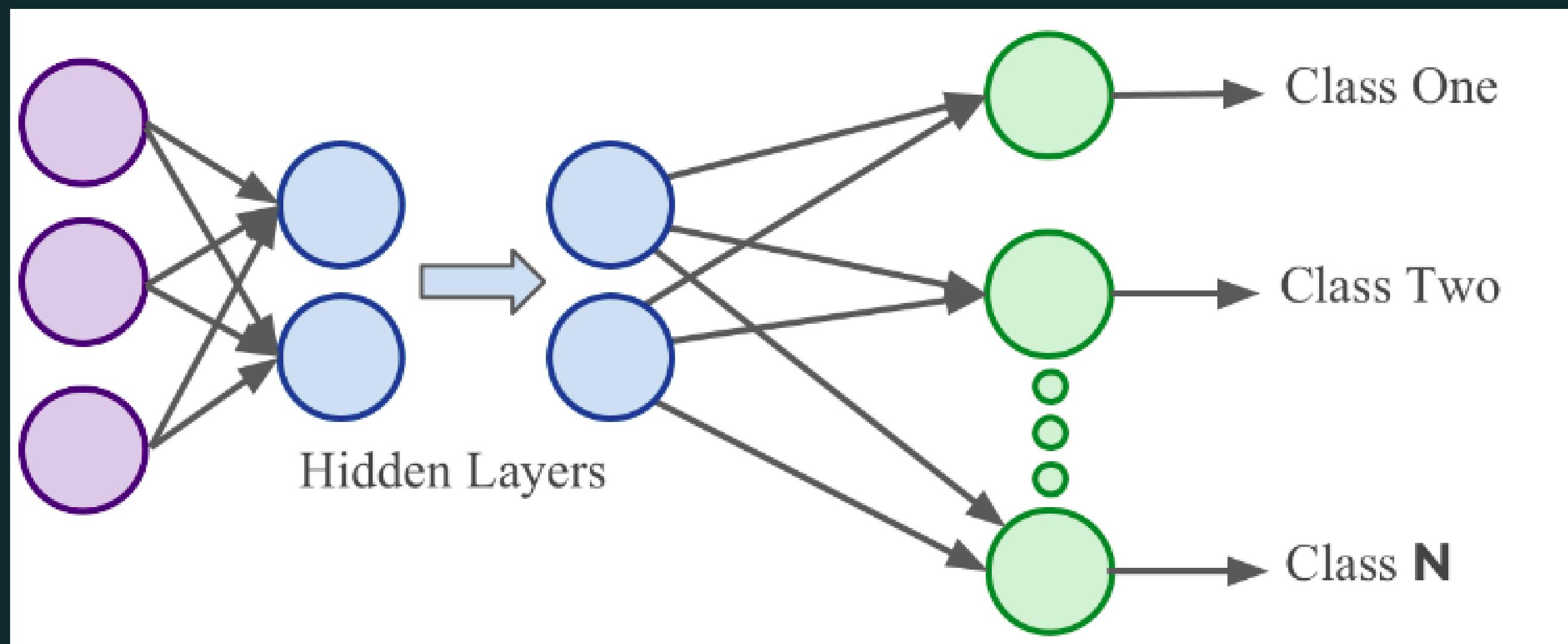


Multi class có thể chia thành 2 trường hợp chính

Mỗi điểm dữ liệu có thể thuộc một hoặc nhiều class

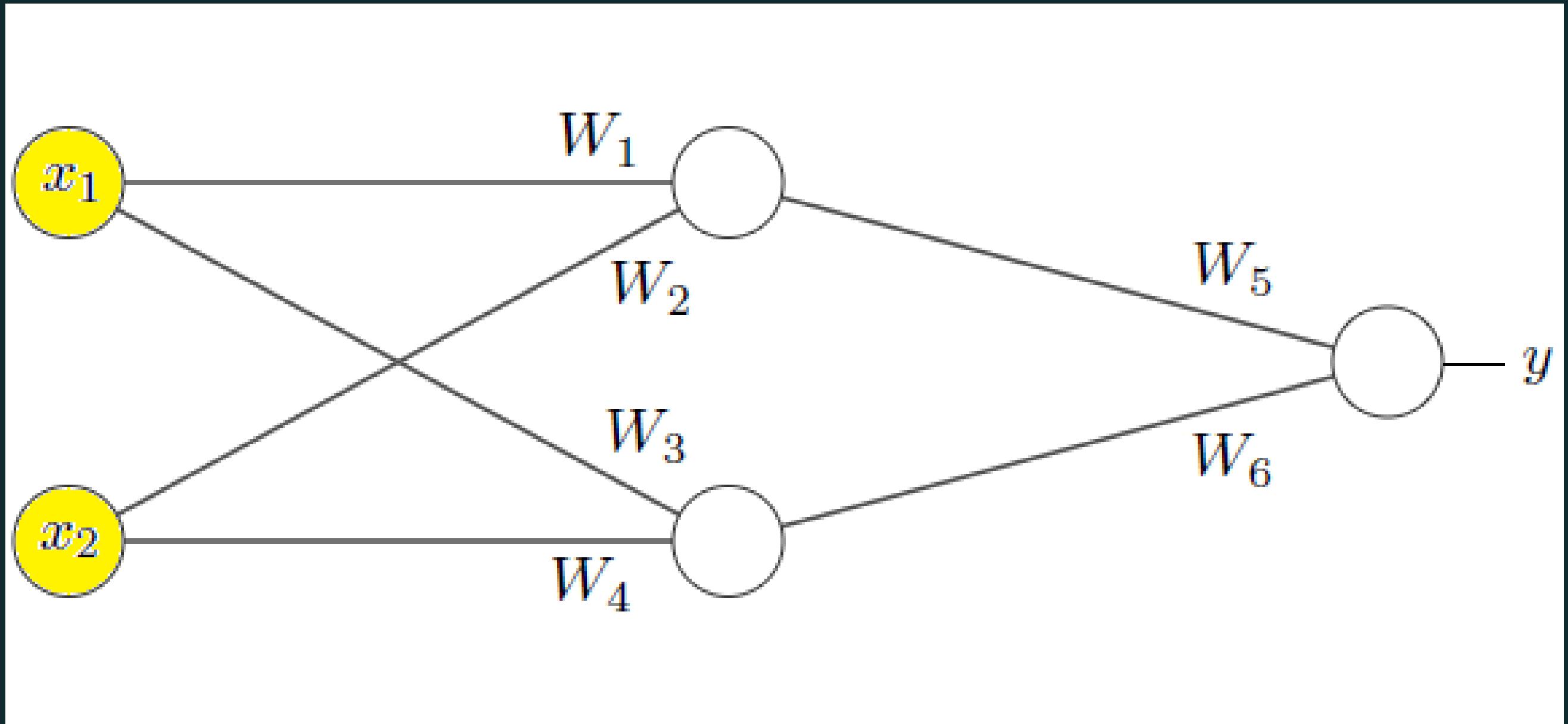


Cách đơn giản nhất là mỗi class sẽ có 1 node output tương ứng và sử dụng các hàm activation phù hợp



Cost function & Gradient Descent

Cost function



<https://www.researchgate.net/publication/332932489/figure/fig1/AS:756125788020736@1557285844270>

Mean Square Error

Một trong những hàm loss khá đơn giản và thường được sử dụng trong các bài toán regression

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

Loss function

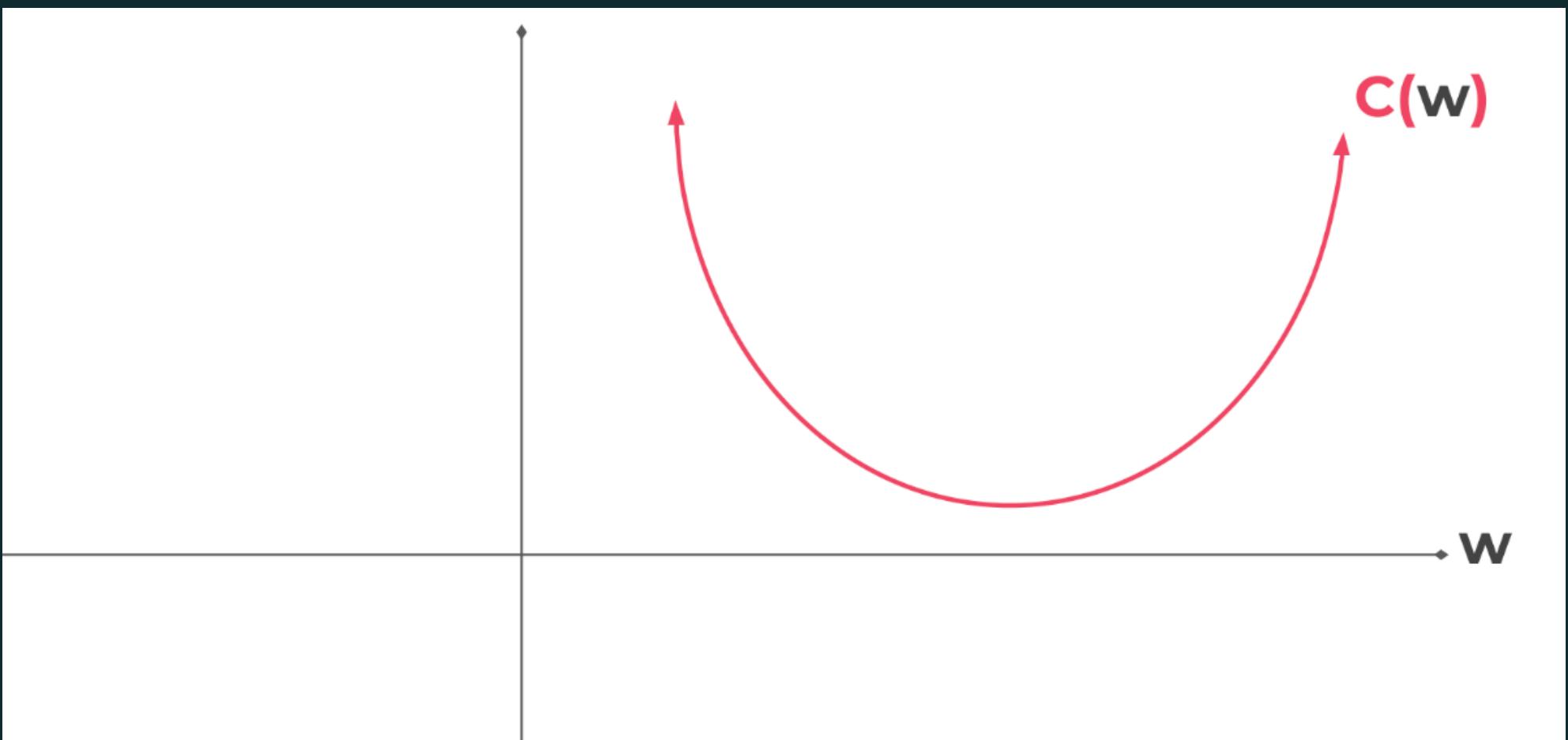
Ngoài ra, còn nhiều hàm loss khác

- Mean Squared Logarithmic Error Loss
- Mean Absolute Error Loss
- Binary Cross-Entropy
- Hinge Loss
- Multi-Class Cross-Entropy Loss
- Focal loss
-

Chúng ta đã biết dùng loss function để đánh giá độ tốt của mạng hình ứng với bộ trọng số (weights)

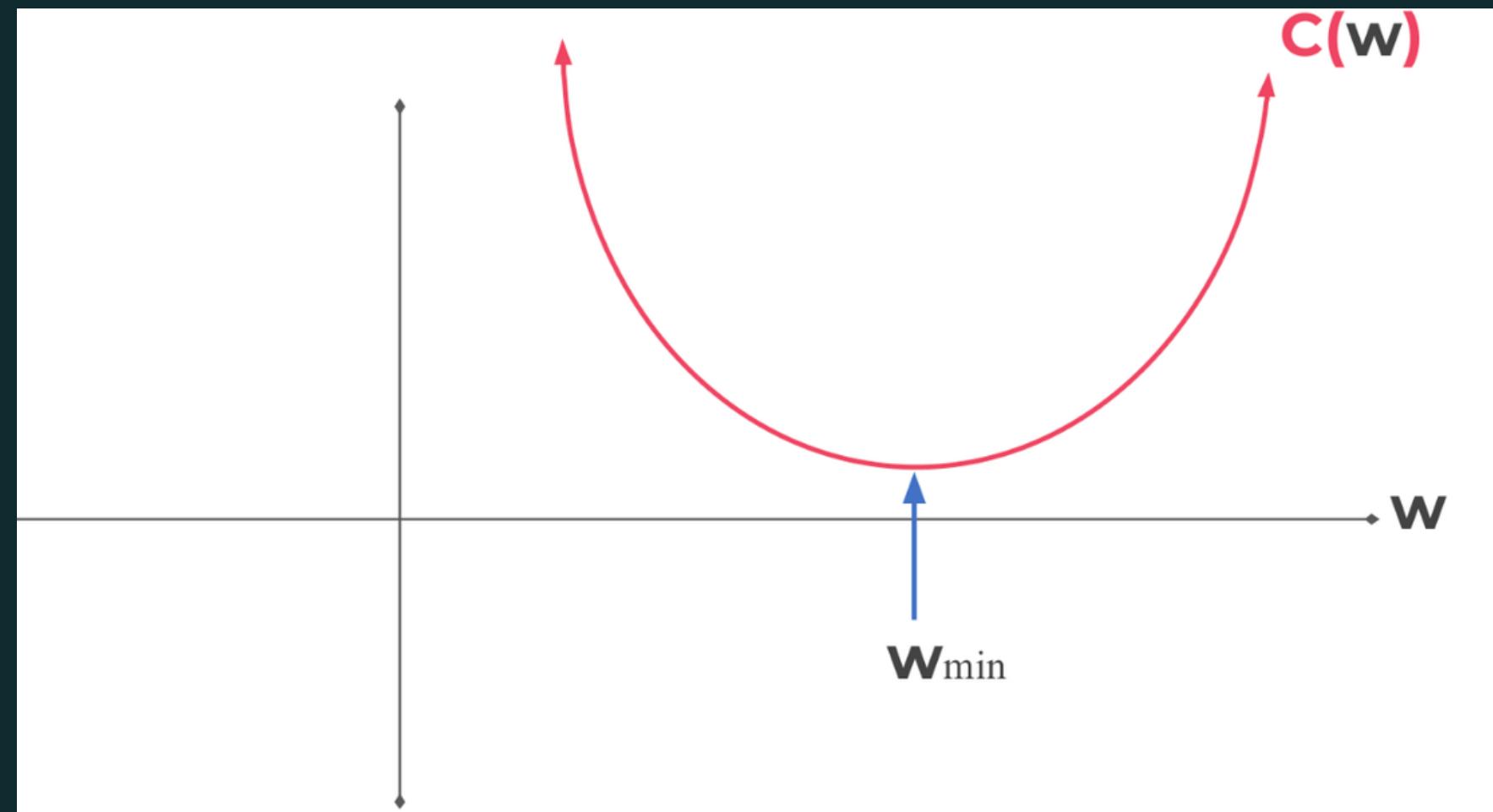
Làm sao để
tìm weights
tốt nhất ?

Để đơn giản, chúng ta có thể dùng
một mạng đơn giản, $L = C(w)$



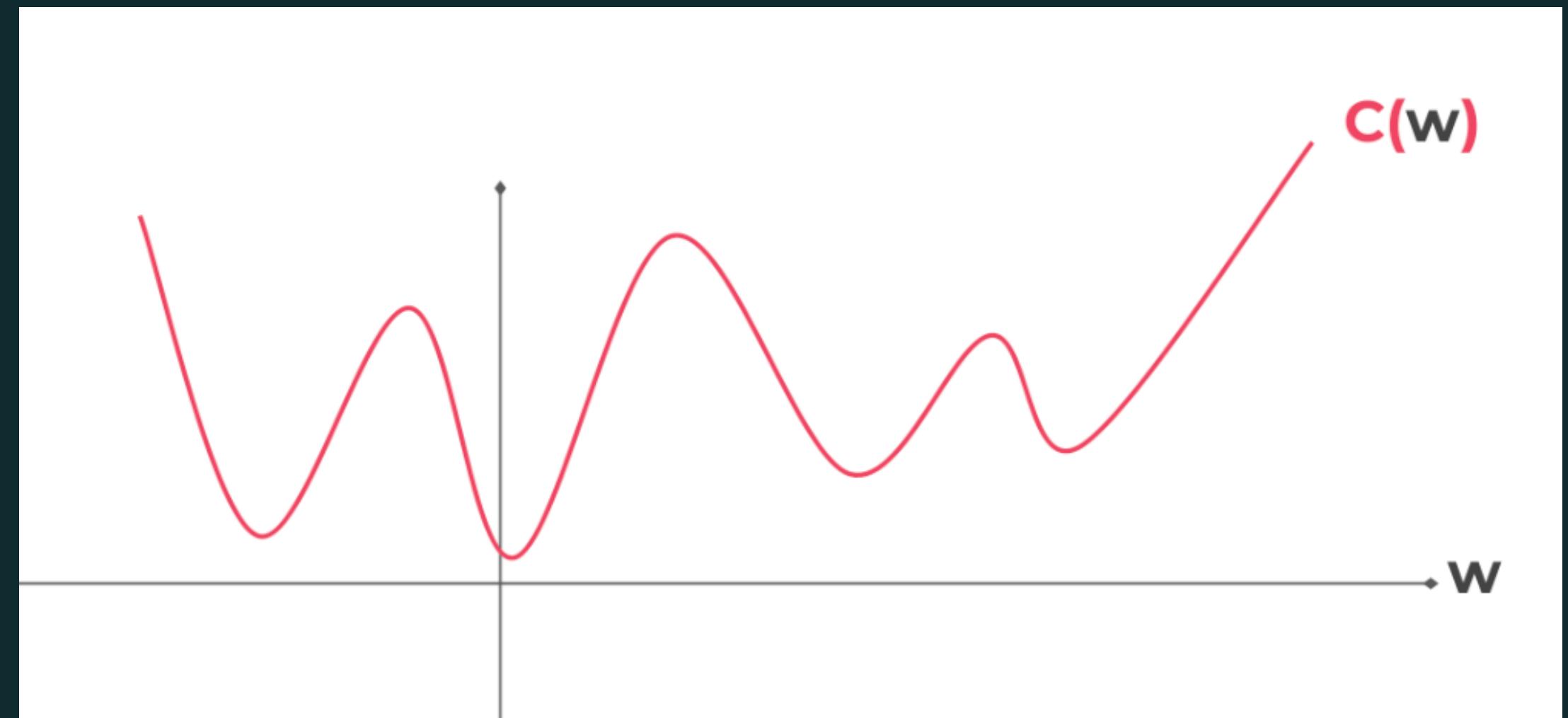
Để đơn giản, chúng ta có thể dùng
một mạng đơn giản, $L = C(w)$

Ta có thể tìm w tốt
nhất bằng cách
tính đạo hàm của
 C , rồi giải đạo
hàm bằng 0

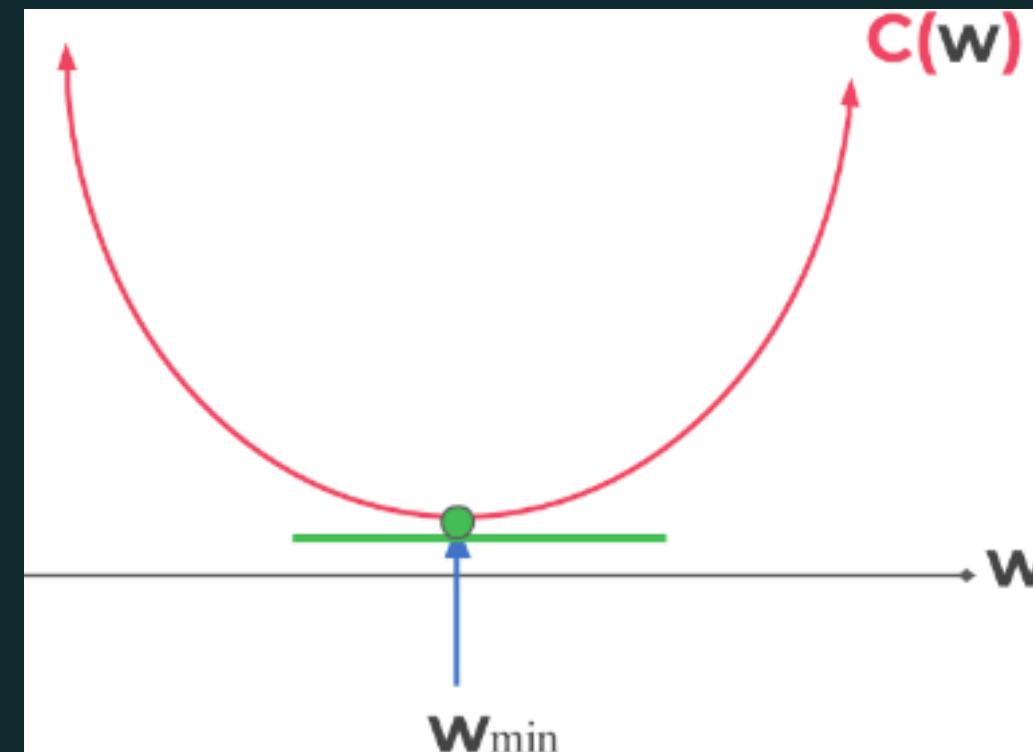
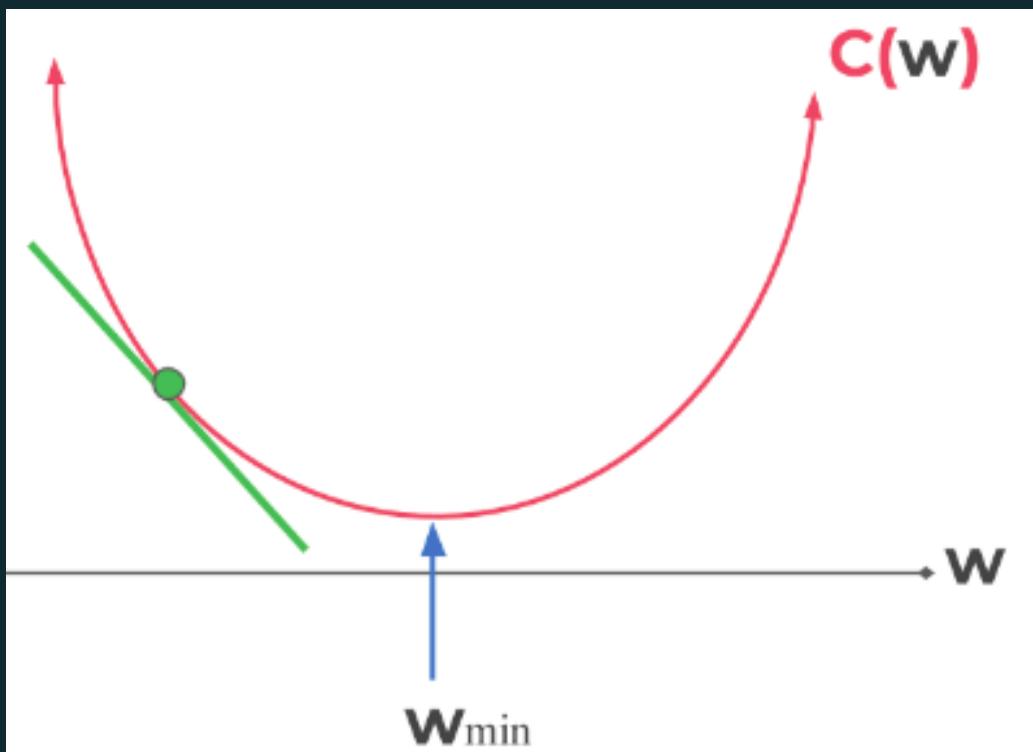
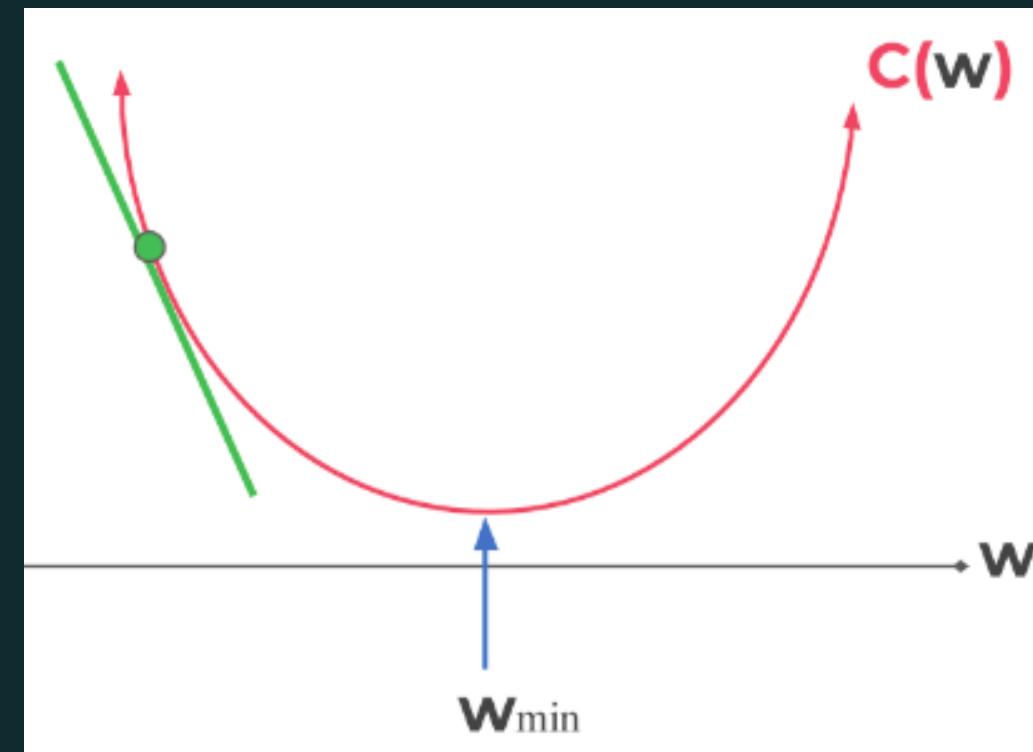
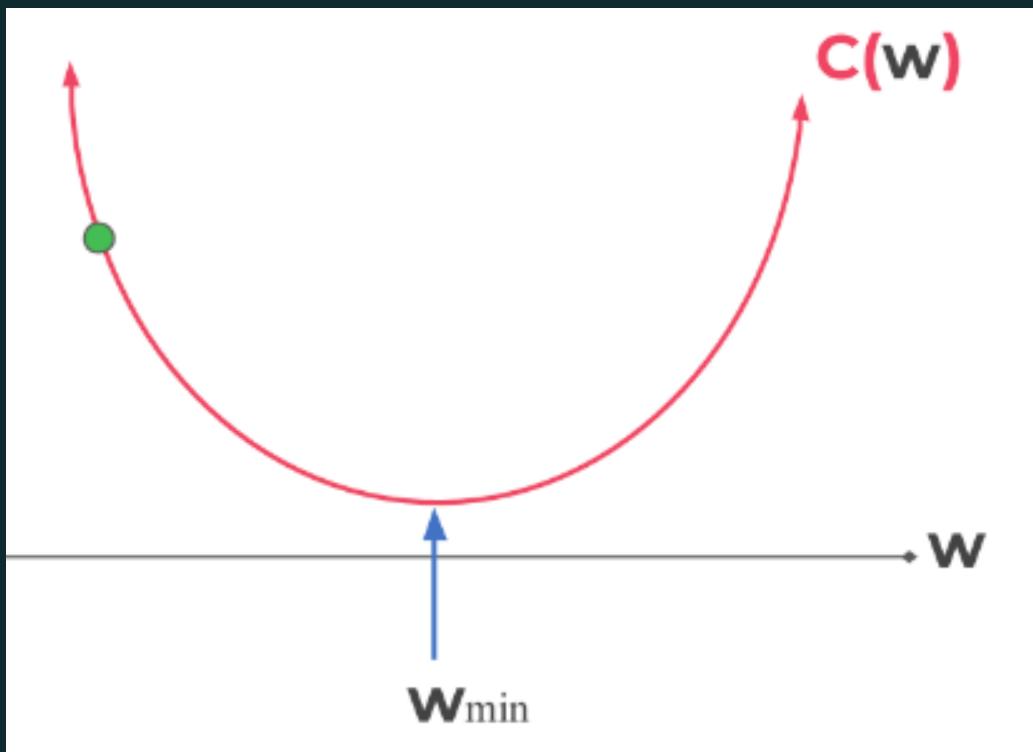


NHƯNG !!!

$C(w)$ có thể là hàm
rất phức tạp và
nhiều chiều

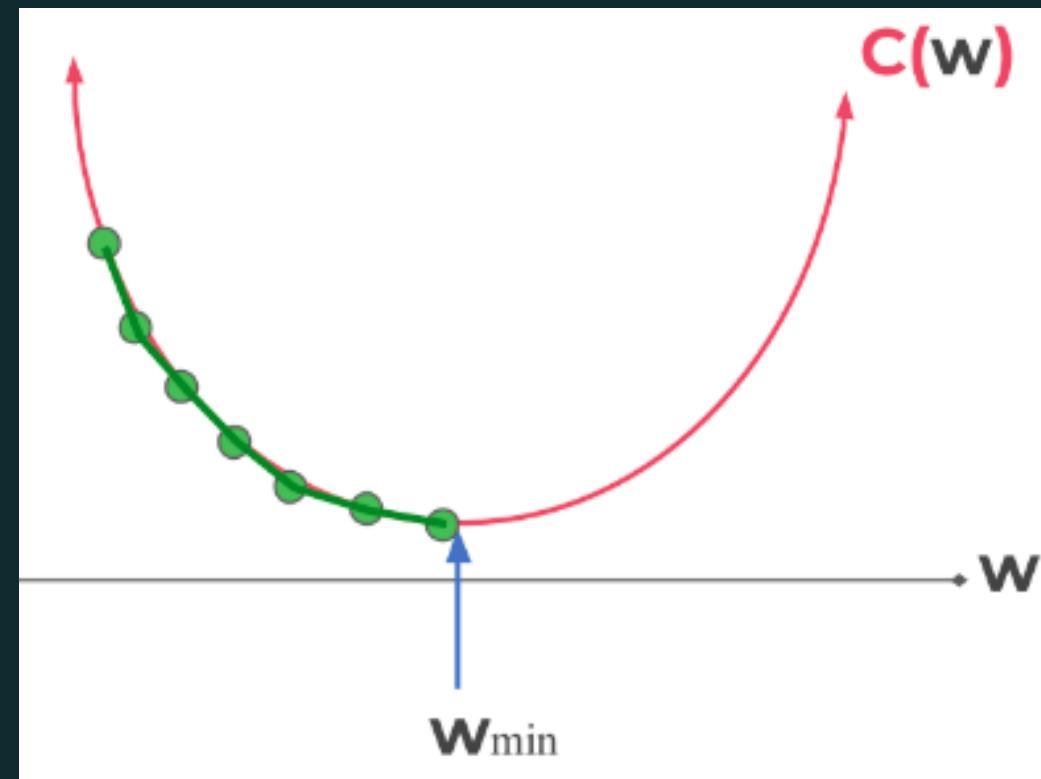


Gradient Descent

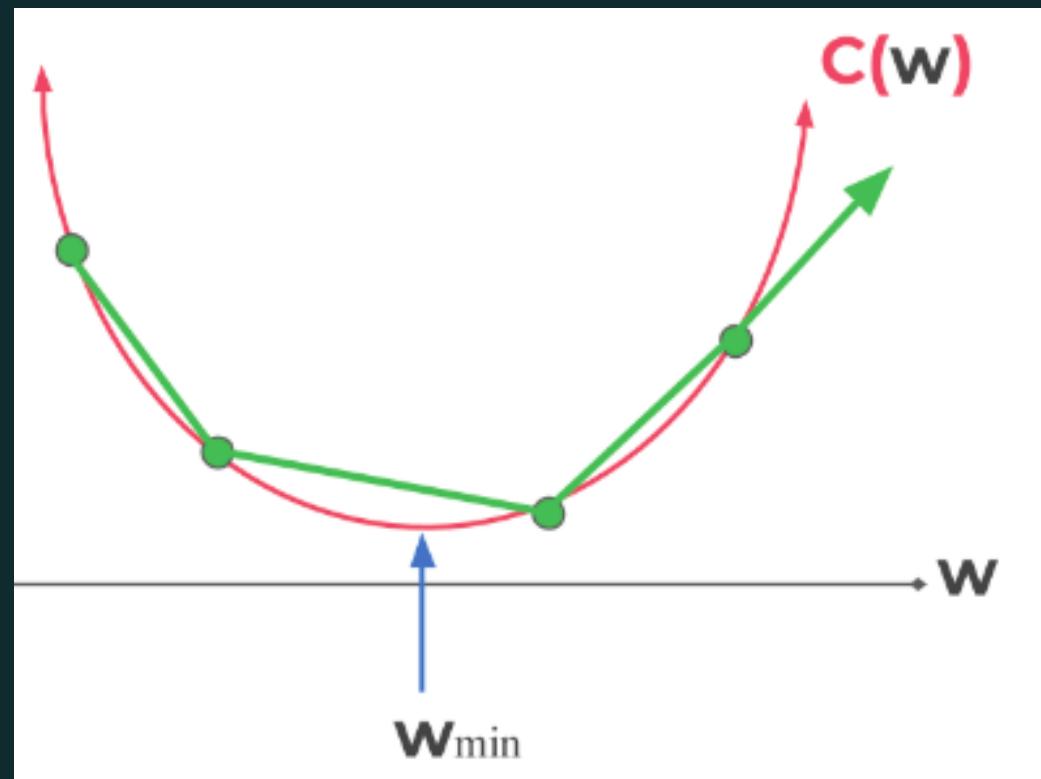


Gradient Descent

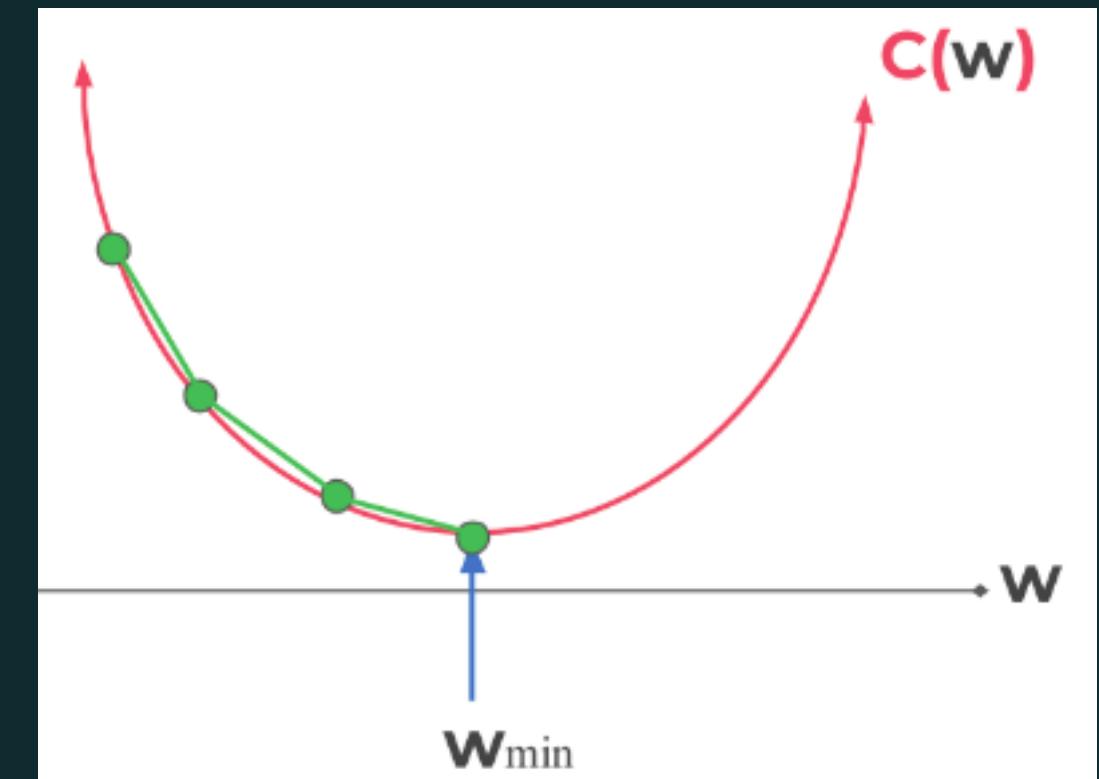
Ta có thể chọn bước di chuyển lớn hay nhỏ cách thiết lập hệ số học (learning rate)



learning rate nhỏ



learning rate lớn



learning rate vừa phải

Backpropagation

Về cơ bản, chúng ta muốn biết cost function thay đổi như thế nào ứng với sự thay đổi của weights, từ đó mới có thể update weight như mong đợi

Giả sử ta có mạng đơn giản



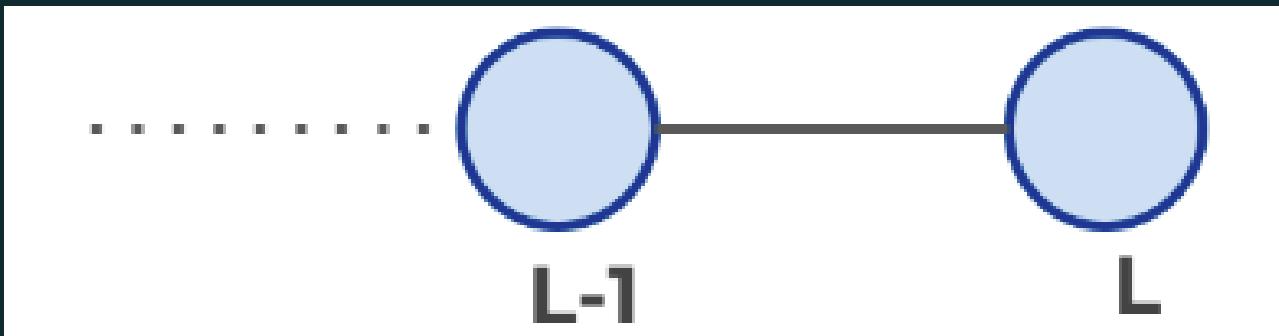
Backpropagation

Xét 2 layer cuối

$$z^L = w^L a^{L-1} + b^L$$

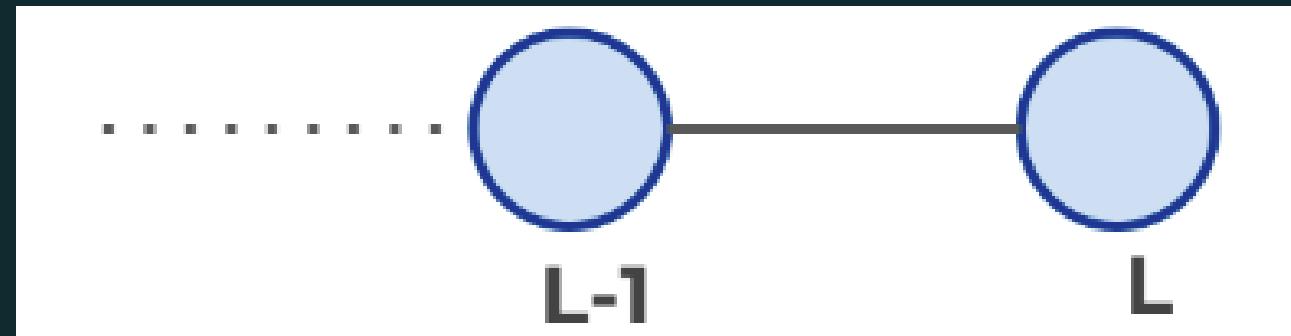
$$a^L = \sigma(z^L)$$

$$C_0(\dots) = (a^L - y)^2$$



Backpropagation

Xét 2 layer cuối



Để biết được sự ảnh hưởng của việc thay đổi w , b lên cost function:

$$\frac{\partial C_0}{\partial w^L} = \frac{\partial z^L}{\partial w^L} \frac{\partial a^L}{\partial z^L} \frac{\partial C_0}{\partial a^L}$$

$$\frac{\partial C_0}{\partial b^L} = \frac{\partial z^L}{\partial b^L} \frac{\partial a^L}{\partial z^L} \frac{\partial C_0}{\partial a^L}$$

Application of ANN

ANN đã trở lên thông dụng và là giải pháp cho nhiều problems: classification, clustering, regression, pattern recognition, dimension reduction, structured prediction, machine translation, anomaly detection, decision making, visualization, computer vision

1. Text Classification and Categorization

- Dùng trong web searching, lọc thông tin, đánh giá khả năng đọc (readability assessment), Phân tích cảm xúc (sentiment analysis)
- Convolutional Neural Networks for Sentence Classification by Yoon Kim: được xây dựng trên CNN và word2Vec,
- Text Understanding from Scratch by Xiang Zhang and Yann LeCun: Dùng deep learning để hiểu text từ ký tự đầu vào, Model được test trên DBpedia ontology classification data set với 14 classes (company, educational institution, artist, athlete, office holder, mean of transportation, building, natural place, village, animal, plant, album, film, written work) với độ chính xác trainning: 99,96% và test: 98.4%

2. Named Entity Recognition (NER)

- NER là phân loại những named entity (như London, Chelsea, Ho Chi Minh, Apple...) thành những category định nghĩa sẵn như người, địa danh, công ty...
- Neural Architectures for Named Entity Recognition: Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, Chris Dyer
- Bài báo sử dụng những bộ data test CoNLL-2002 và CoNLL-2003 ở ngôn ngữ tiếng Anh, Đức, Tây Ban Nha, Đan Mạch và kết luận rằng: không cần có kiến thức về ngôn ngữ cụ thể hoặc một từ điển địa lý, model vẫn có performance tốt

3. Semantic Parsing and Question Answering

- Hệ thống trả lời câu hỏi sẽ tự động trả lời các câu hỏi trong ngôn ngữ tự nhiên
- Semantic Parsing via Staged Query Graph Generation Question Answering with Knowledge Base Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao
- Model được test trên dataset WebQuestions và trả về kết quả tốt

4. Speech Recognition

- Ứng dụng trong lĩnh vực nhà tự động, trợ lý ảo (Siri, Cotana ...), video game.
- Convolutional Neural Networks for Speech Recognition của các tác giả Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu dùng mạng CNN để nhận dạng giọng nói theo một cách mới, sao cho cấu trúc của CNN trực tiếp điều chỉnh một số dạng biến đổi giọng nói như tốc độ nói khác nhau và tìm kiếm bằng giọng nói có lượng từ vựng lớn đã được sử dụng.

4. Speech Recognition

- Ứng dụng trong lĩnh vực nhà tự động, trợ lý ảo (Siri, Cotana ...), video game.
- Convolutional Neural Networks for Speech Recognition của các tác giả Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu dùng mạng CNN để nhận dạng giọng nói theo một cách mới, sao cho cấu trúc của CNN trực tiếp điều chỉnh một số dạng biến đổi giọng nói như tốc độ nói khác nhau và tìm kiếm bằng giọng nói có lượng từ vựng lớn đã được sử dụng.



Demo



Simple Image Classification using ANN

Labels

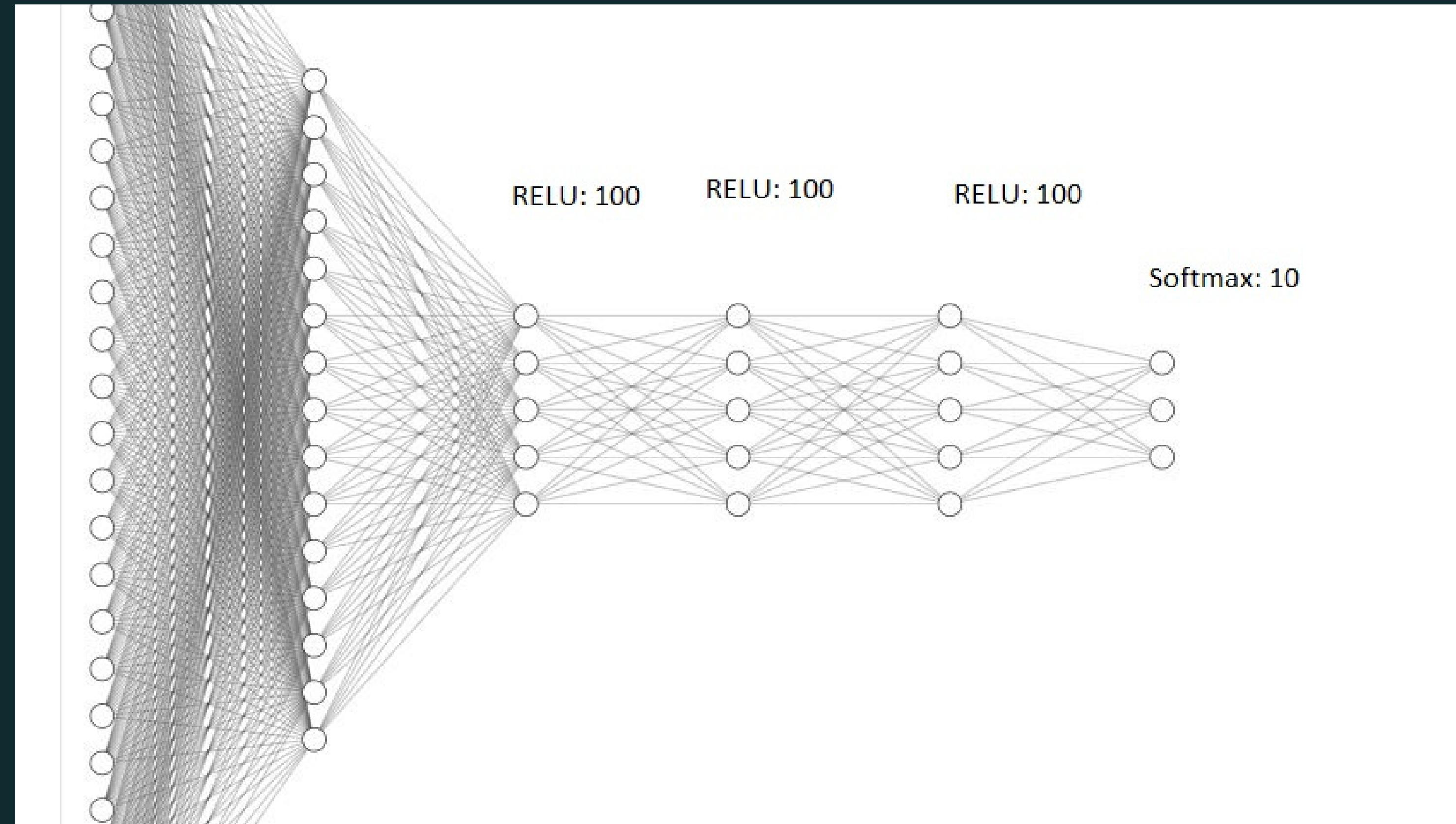


Tranning Dataset

MNIST là 1 dataset chứa chữ số viết tay, hình ảnh với kích thước cố định tập trainning có 60,000 mẫu và tập test có 10,000 mẫu.

<http://yann.lecun.com/exdb/mnist/>

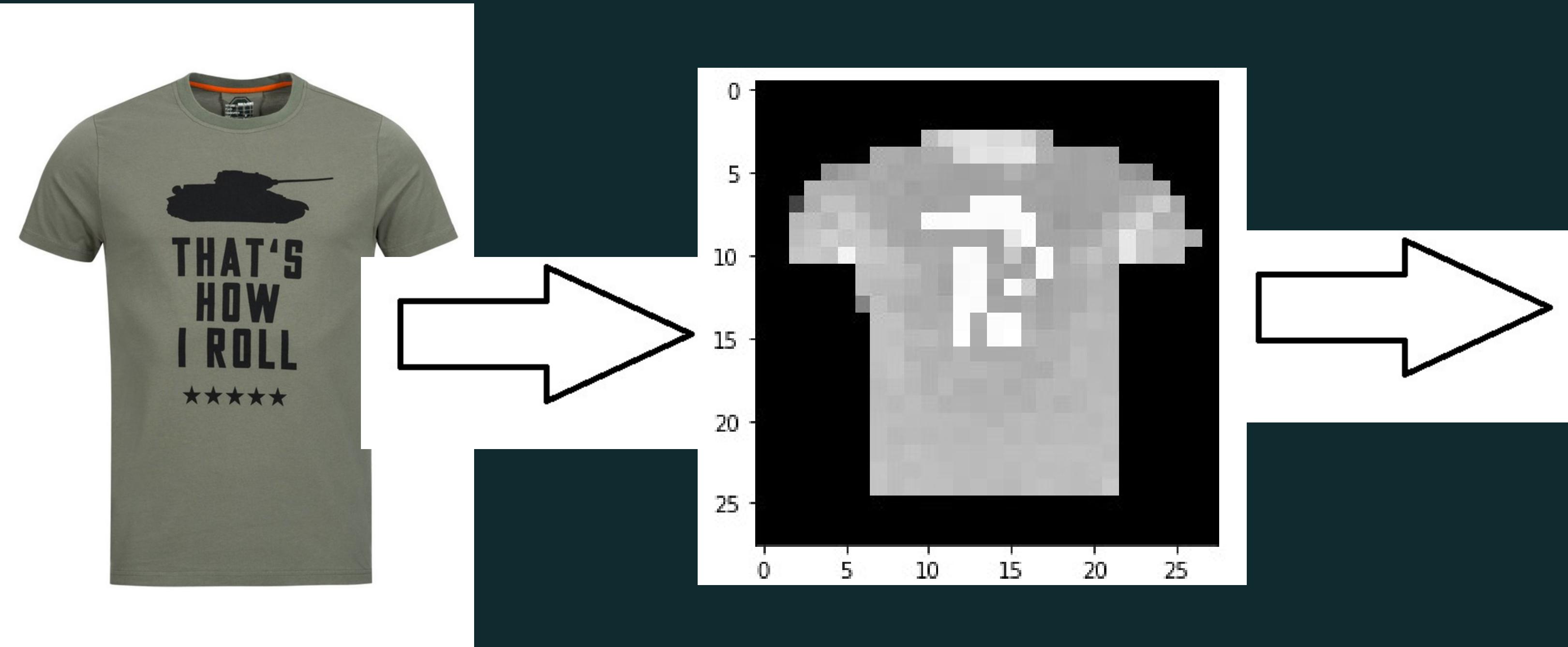
Network architecture



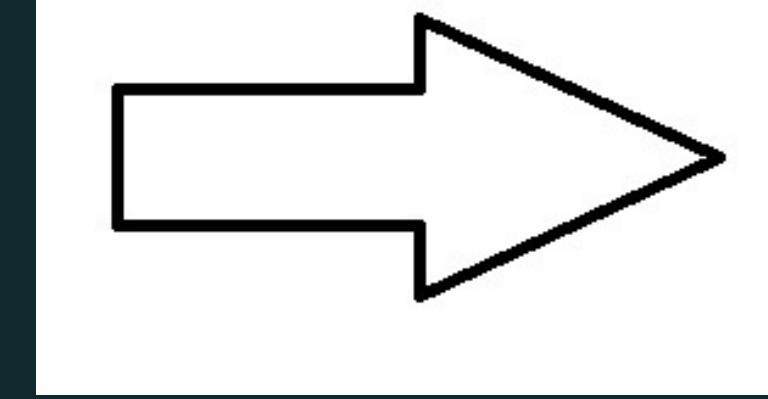
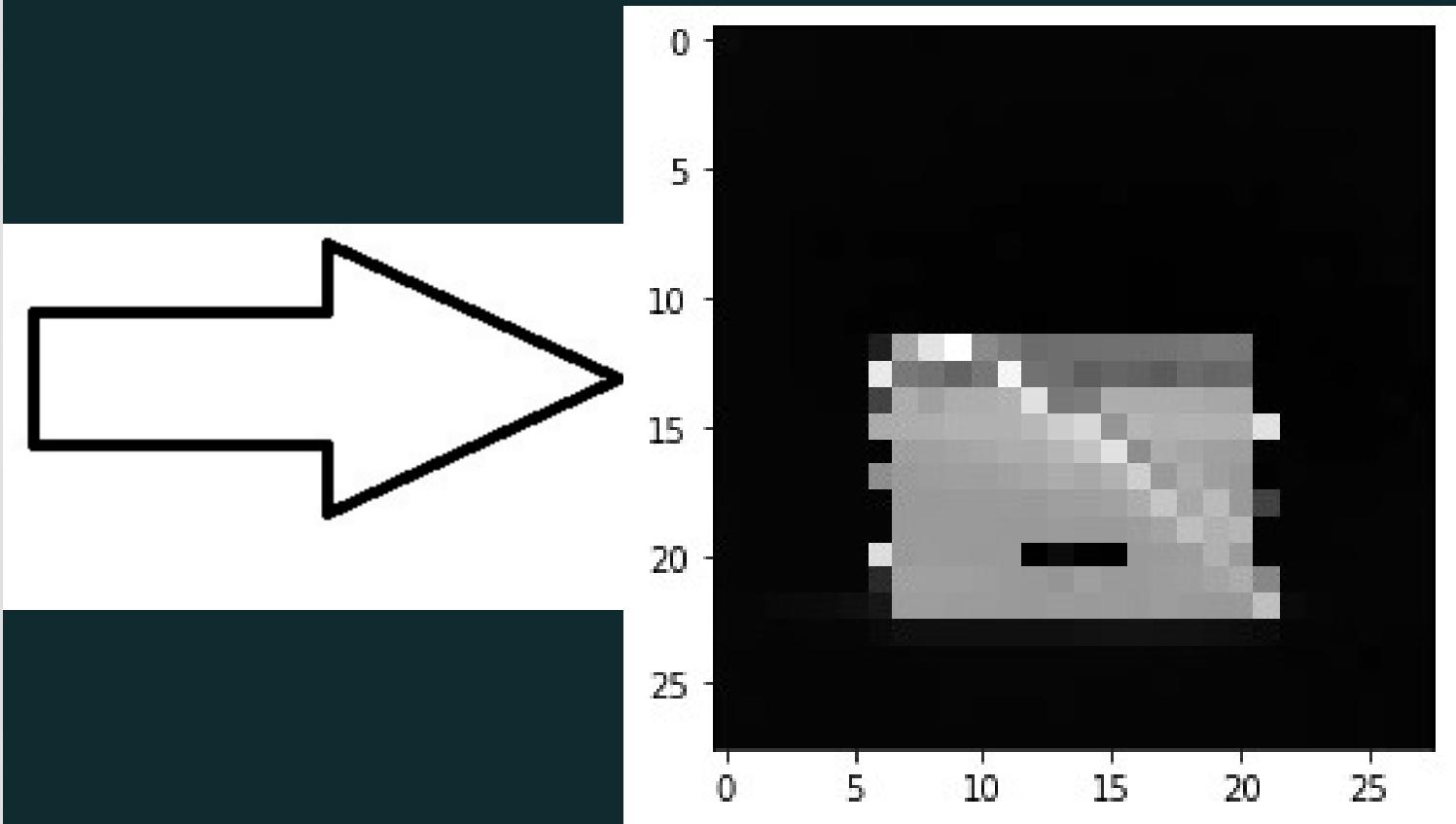
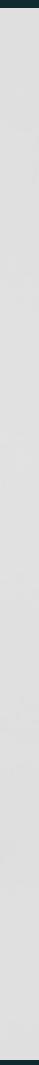
Network architecture

- Input node: $28 \times 28 = 784$
- Hidden layer 1 (ReLU): 300
- Hidden layer 2 (ReLU): 100
- Hidden layer 3 (ReLU): 100
- Hidden layer 4 (ReLU): 100
- Output layer (Softmax): 10

Test model



Test model



Bag

Tài liệu tham khảo

- <https://medium.com/analytics-steps/understanding-the-perceptron-model-in-a-neural-network-2b3737ed70a2>
- <https://youtu.be/bfmFfD2RIcg>
- <https://medium.com/nerd-for-tech/how-to-train-neural-networks-for-image-classification-part-1-21327fe1cc1>
- <https://medium.com/@datamonsters/artificial-neural-networks-in-natural-language-processing-bcf62aa9151a>
- <https://aclanthology.org/D14-1181/>
- <https://arxiv.org/pdf/1502.01710.pdf>
- <https://aclanthology.org/N16-1030/>
- <https://aclanthology.org/P15-1128/>
- <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/CNN ASLPTTrans2-14.pdf>

Tài liệu tham khảo

- <https://medium.com/analytics-steps/understanding-the-perceptron-model-in-a-neural-network-2b3737ed70a2>
- <https://youtu.be/bfmFfD2RIcg>