

Introduction to Neural Networks



Outline

- Introduction to neural networks
- Types of neural networks: feedforward, convolutional, and recurrent
- Activation functions
- Building a simple neural network using Keras

Introduction to Neural Networks

Membuat Gambar dari tulisan



“Creating a man that rides motorcycles on Mars.”

Membuat tulisan dengan chat gpt



Jaringan saraf atau neural networks adalah model komputasi yang terinspirasi oleh struktur dan fungsi otak manusia. Mereka terdiri dari unit pemrosesan kecil yang disebut neuron, yang saling terhubung dan bekerja sama untuk memproses informasi. Neural networks memungkinkan mesin untuk "belajar" dari data dan membuat prediksi atau pengambilan keputusan berdasarkan pola yang terdeteksi dalam data tersebut.

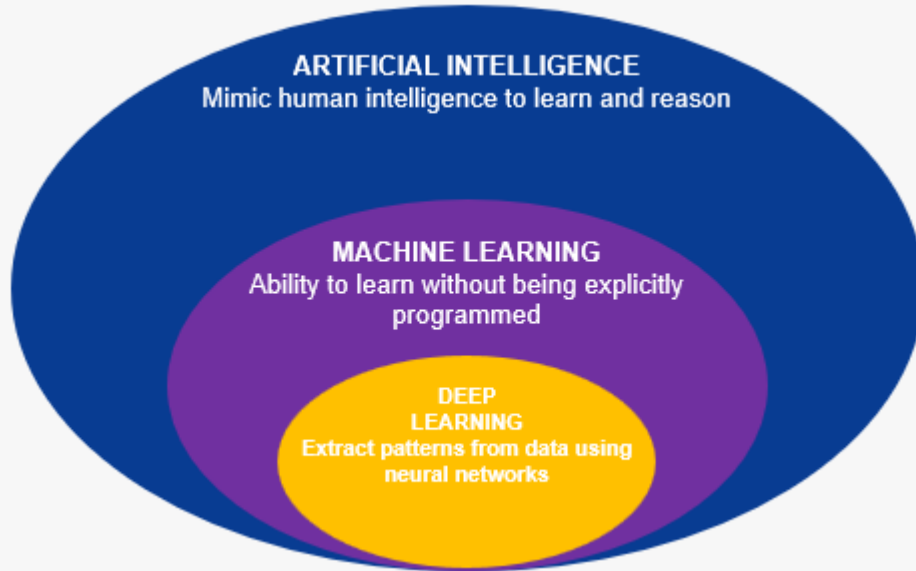
Dalam neural networks, setiap neuron menerima input, menghitung outputnya sendiri, dan mengirimkan output tersebut ke neuron lainnya. Informasi mengalir melalui jaringan ini, melalui koneksi yang kuat antara neuron-neuron, yang disebut bobot. Selama proses pembelajaran, bobot tersebut disesuaikan secara otomatis berdasarkan data pelatihan untuk meningkatkan kemampuan jaringan dalam mempelajari pola yang kompleks.

Keunggulan neural networks adalah kemampuannya untuk menangani masalah yang rumit dan non-linear. Mereka dapat digunakan dalam berbagai bidang, seperti pengenalan pola, klasifikasi data, analisis citra, pengenalan suara, dan prediksi. Neural networks telah menunjukkan keberhasilan yang signifikan dalam aplikasi seperti kendaraan otonom, pengenalan wajah, penerjemahan bahasa, dan rekomendasi produk.

Dalam bahasa yang sederhana, neural networks adalah model komputasi yang meniru cara kerja otak manusia dan digunakan untuk mempelajari pola dan membuat prediksi berdasarkan data. Mereka adalah alat yang kuat dalam pengolahan informasi dan telah digunakan dalam berbagai aplikasi yang membutuhkan pemahaman dan analisis data yang mendalam.

“Buatlah penjelasan singkat tentang neural networks”

Apa itu Neural Network

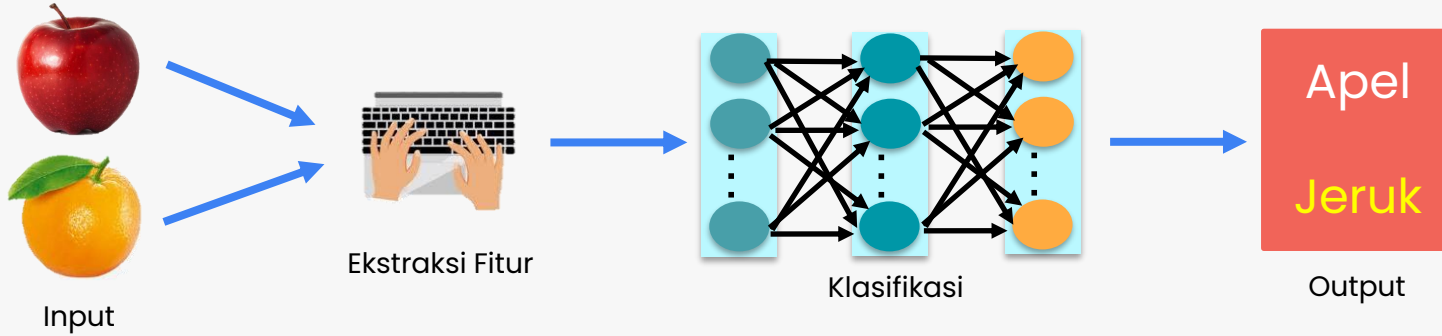


Bagian dalam **Deep Learning**, untuk mengajarkan komputer untuk memproses data dengan cara yang terinspirasi dari otak manusia

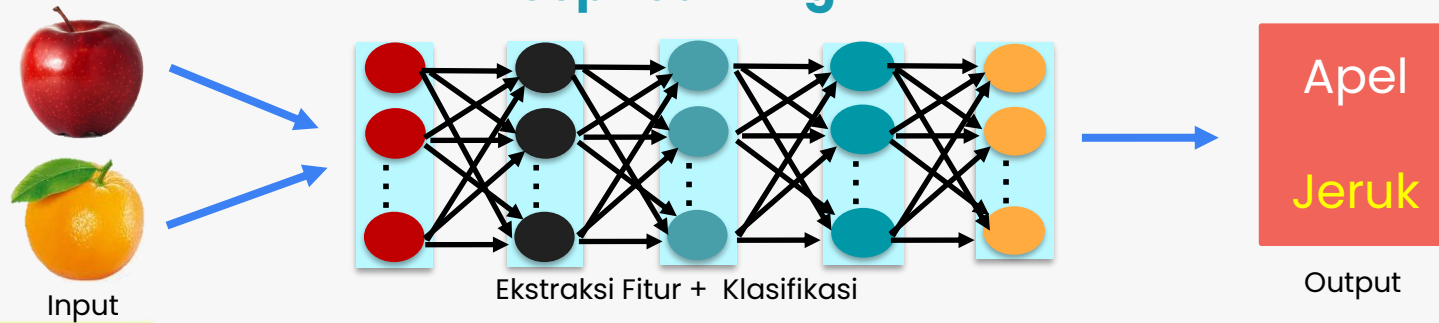
Perkembangan Neural Networks



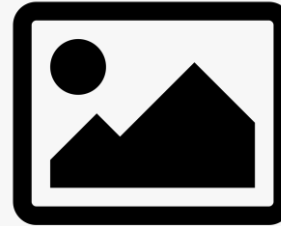
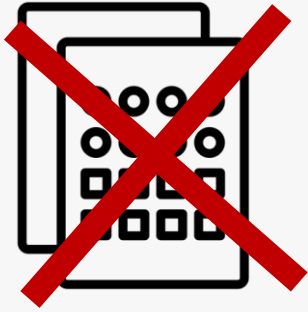
Machine Learning



Deep Learning

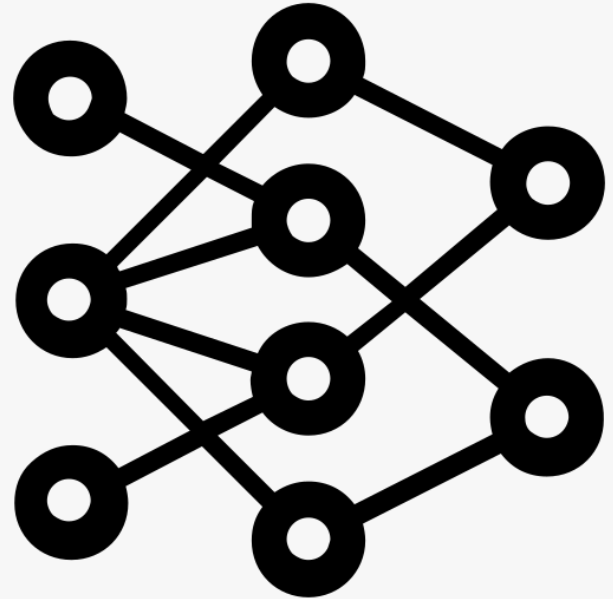


Unstructured Data



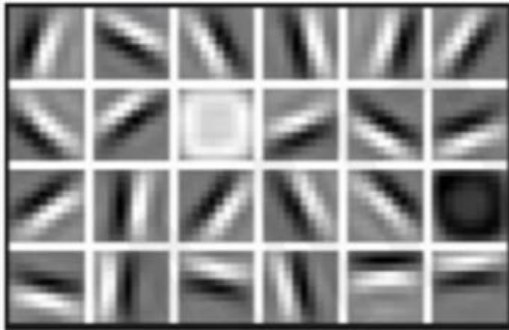
Kapan menggunakan Neural Networks

- Mengatasi data tidak berstruktur, misal Gambar Apel dan Jeruk
- Tidak perlu memusingkan, kenapa neural network tahu outputnya, misal apakah Apel atau Jeruk
- Dapat menentukan arsitektur yang sesuai dalam Neural Networks, misal CNN



Why Neural Network?

Tahapan **feature-engineering** memerlukan waktu yang banyak, dan tidak dapat di skalabilitas.



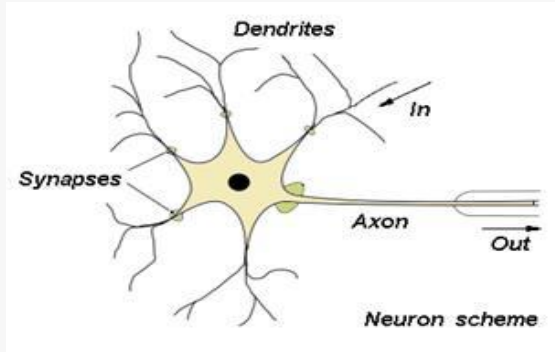
Low level features
Lines & Edges



Mid level features
Eye, Nose, Face

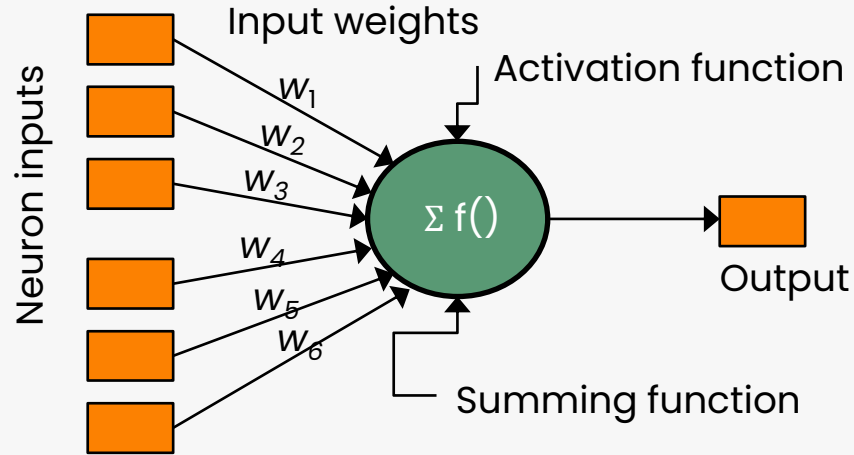


High level features
Facial Structure



Human Neuron

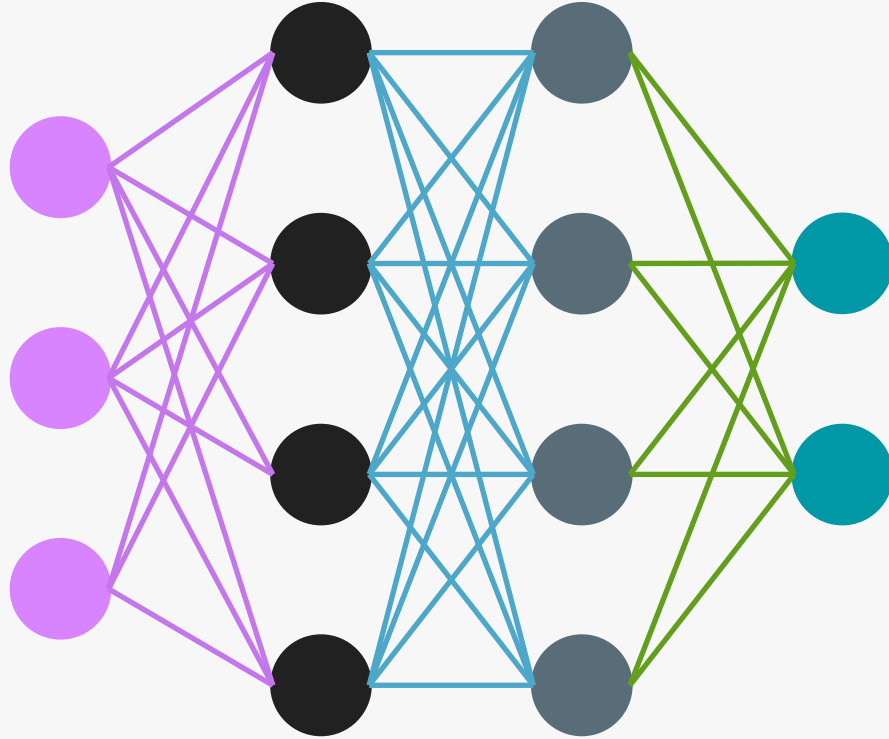
- Multiple Dendrit (*Dendrites*) menerima input
- Nucleus melakukan pemrosesan, dan Sinapsis sebagai fungsional
- Single Axon meneruskan output



Artificial Neuron

- Multiple Input
- Transfer dan Activation Functions
- Single Output

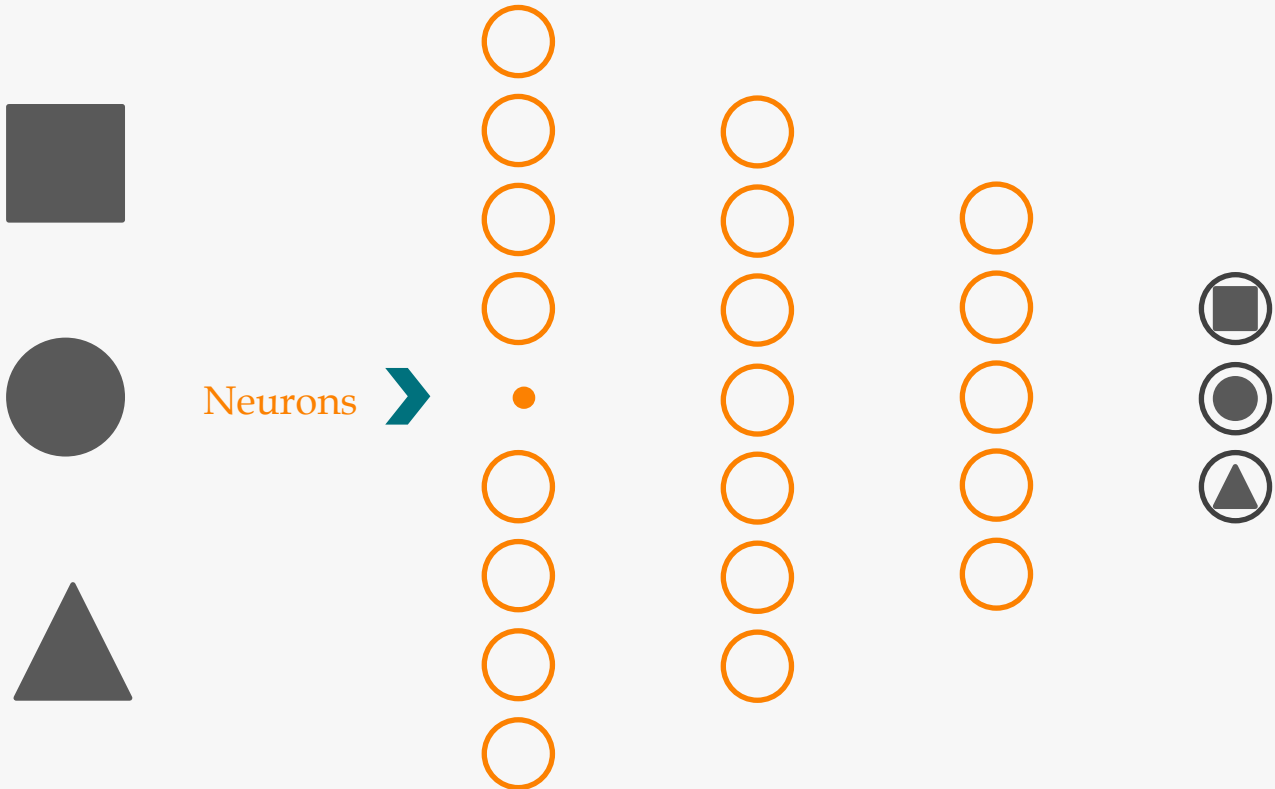
Human Neuron	Artificial Neuron
Soma	Neuron
Dendrite	Input
Axon	Output
Synapse	Weight



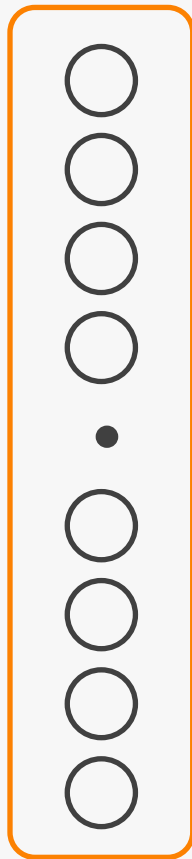
Output

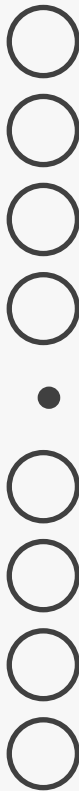
Artificial Neural Network



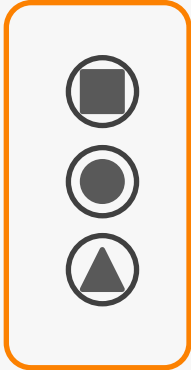


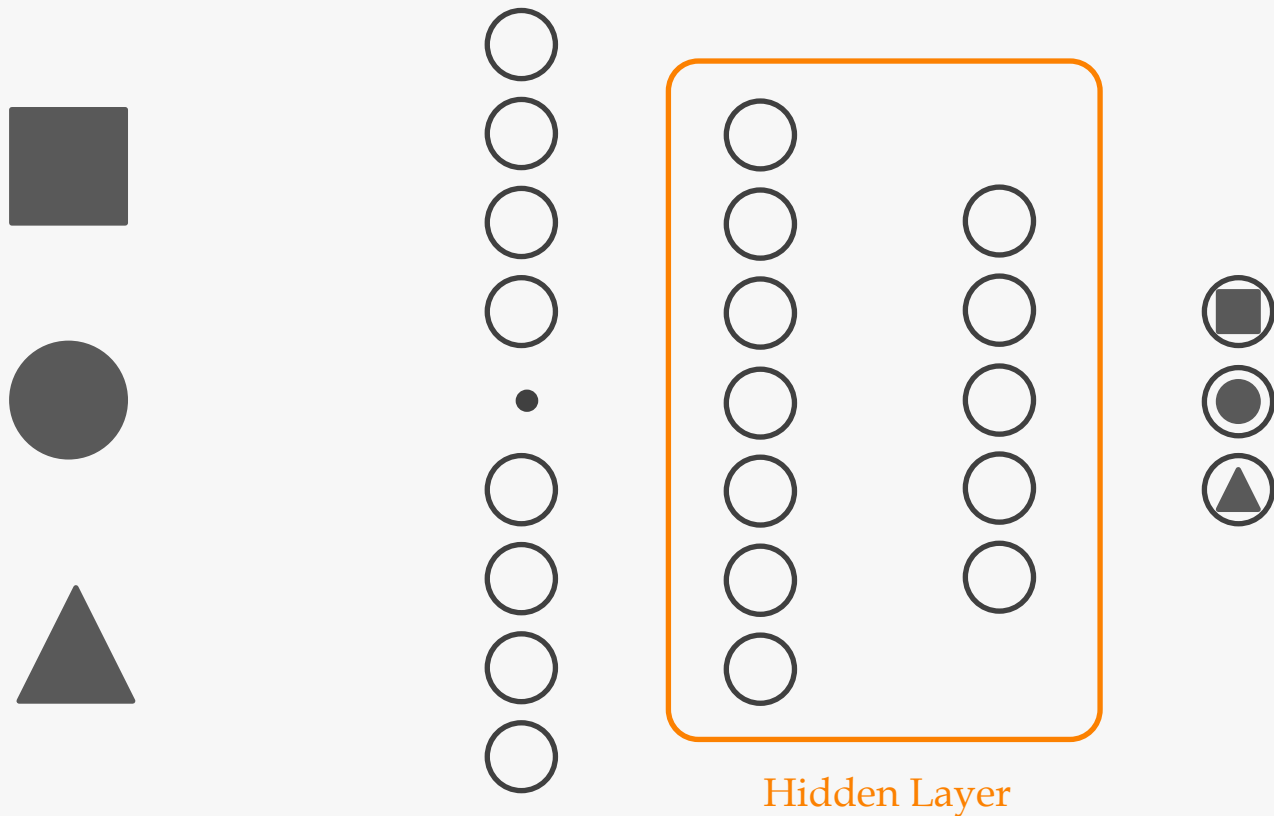
Input Layer



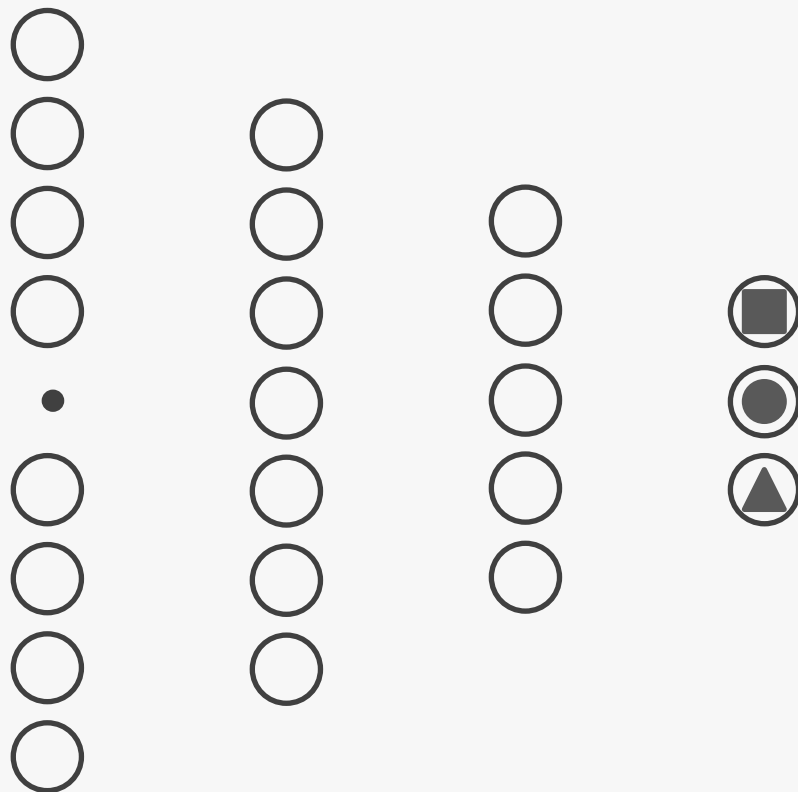
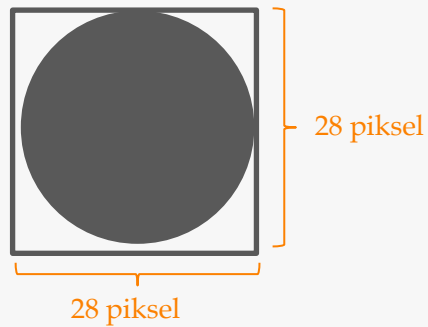


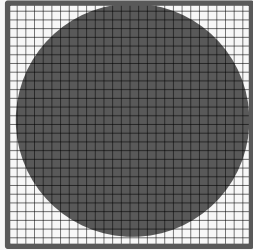
Output Layer









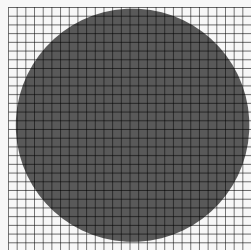


28 piksel

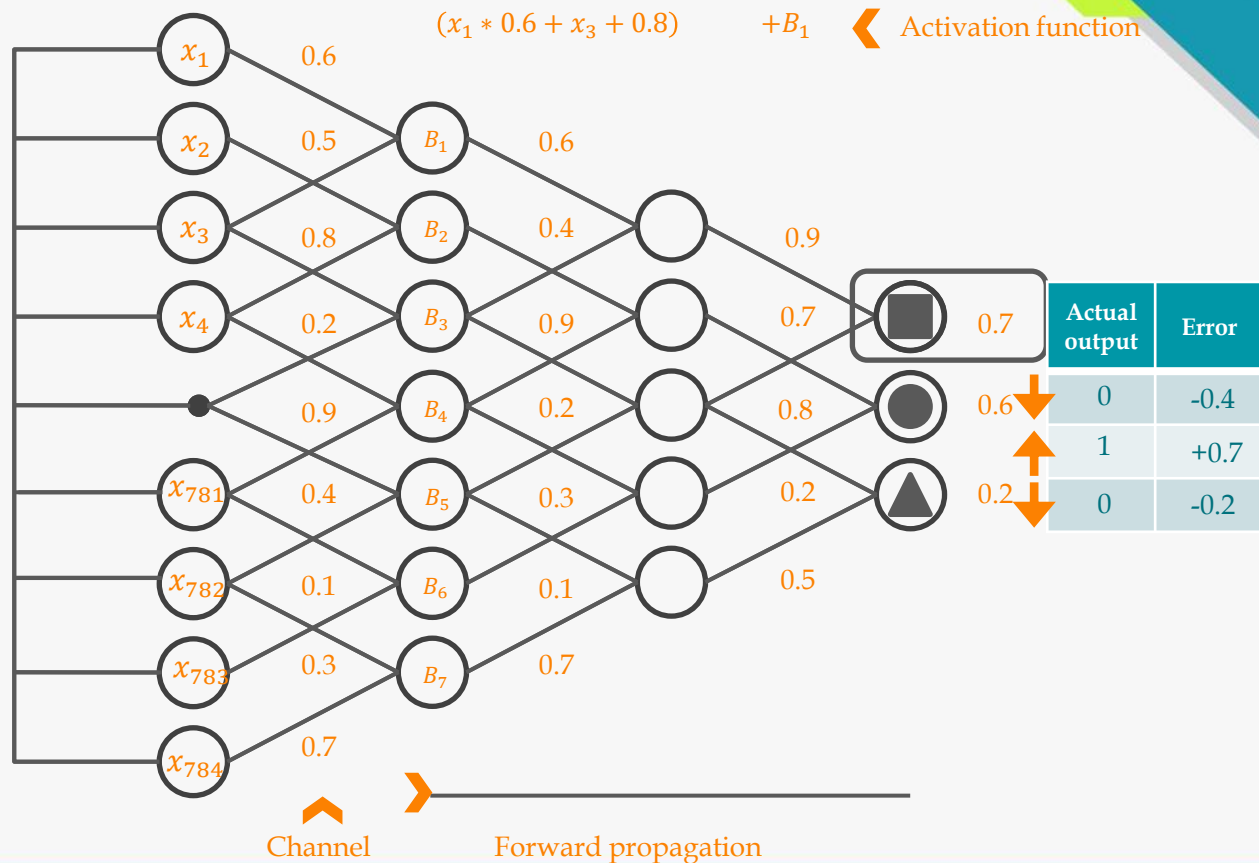
28 piksel

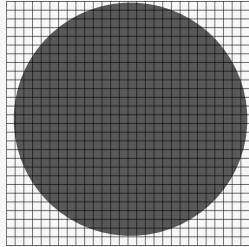
$$28 \times 28 = 784 \text{ pixels}$$



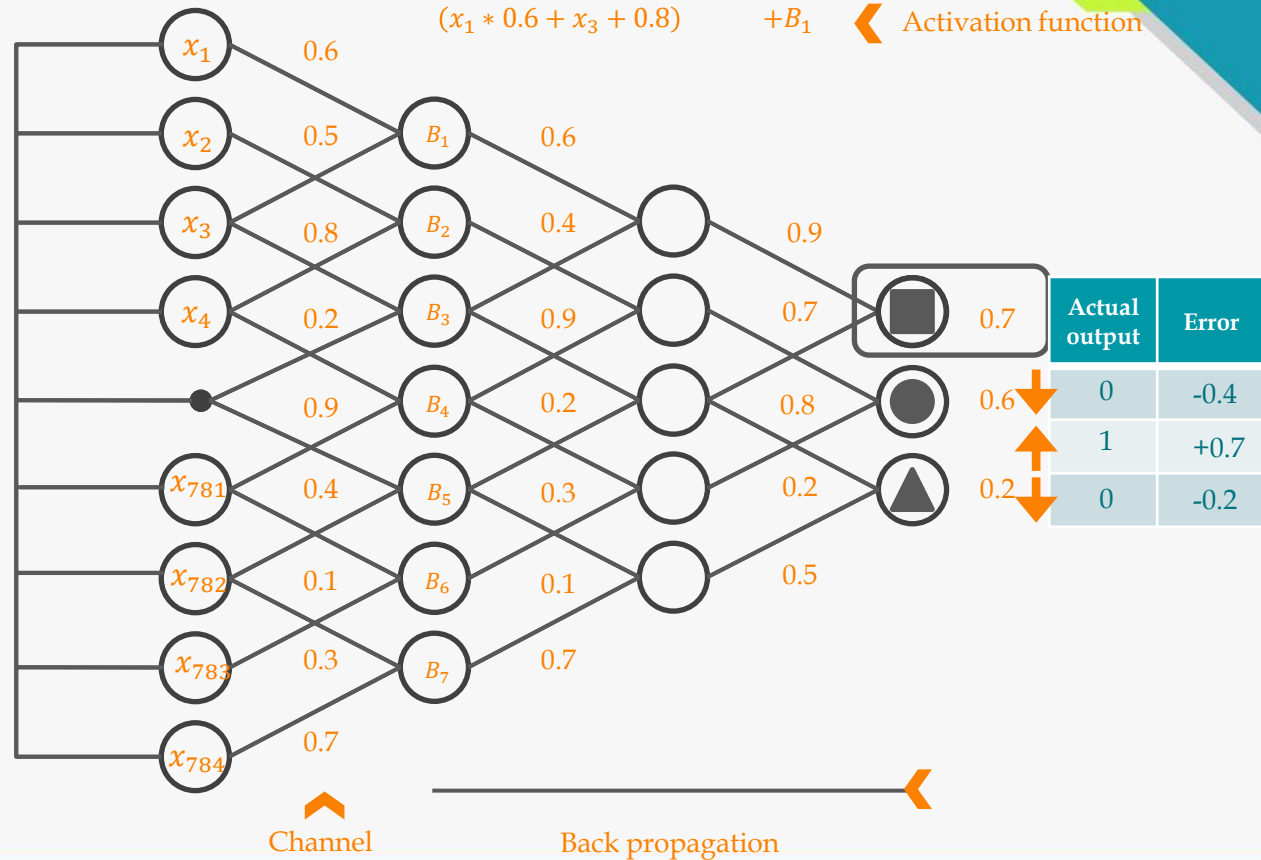


$28 \times 28 = 784$ pixels



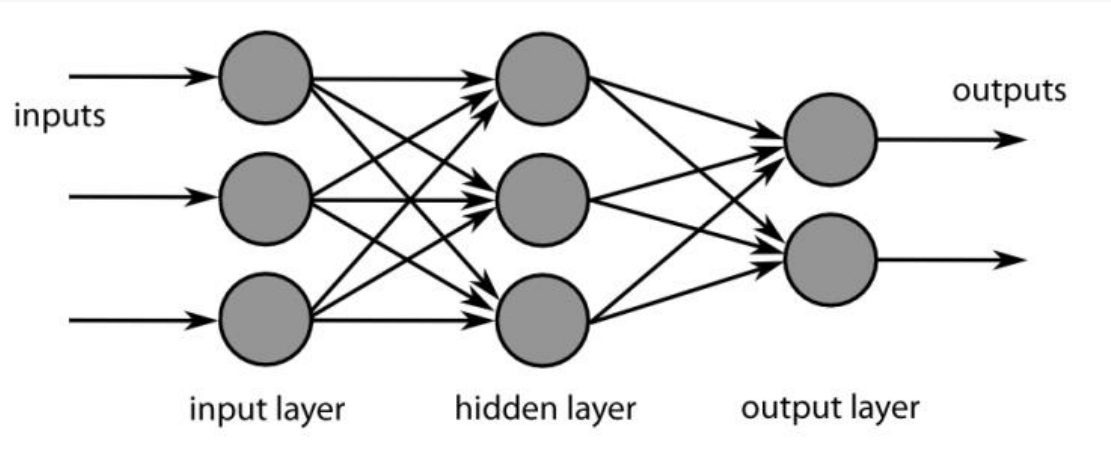


$28 \times 28 = 784$ pixels



Types of neural networks: feedforward, convolutional, and recurrent

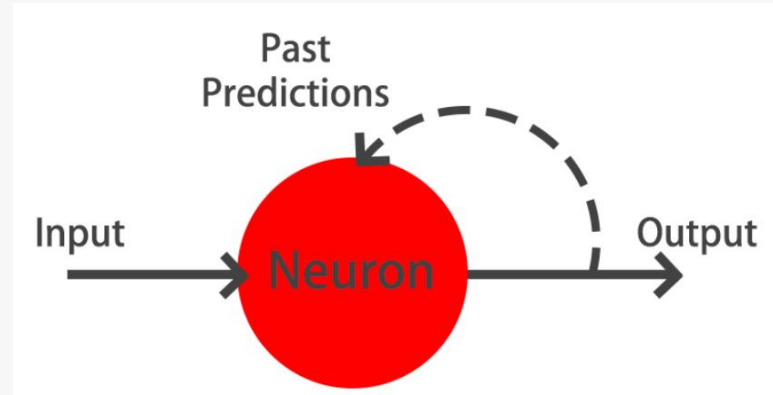
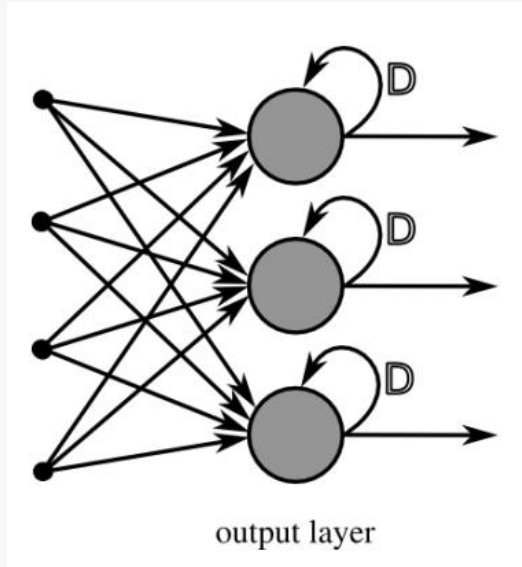
FeedForward



Penggunaan: Apa saja

Tidak cocok untuk: Images, text, time-series.




Recurrent



Penggunaan: Text

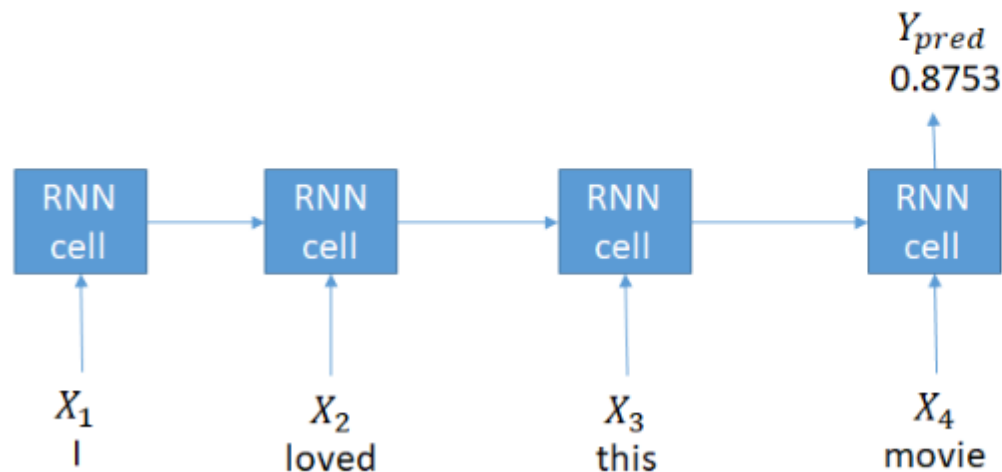


Aplikasi RNN dalam Text

Sentiment Analysis	Multiclass Class	Text Generation	Neural Translation
		<p>David Amore Cecchini to me ▾ Have you heard? Next World Cup is going to be awesome, and you will be rooting for the Seleção, right? Best,</p> <p>Yes! No, I haven't. Not yet!</p> 	<p>English Indonesian</p> <p>Neural Networks</p> <p>Jaringan Neural</p>

Sequence Models (1/3)

Many to one: classification



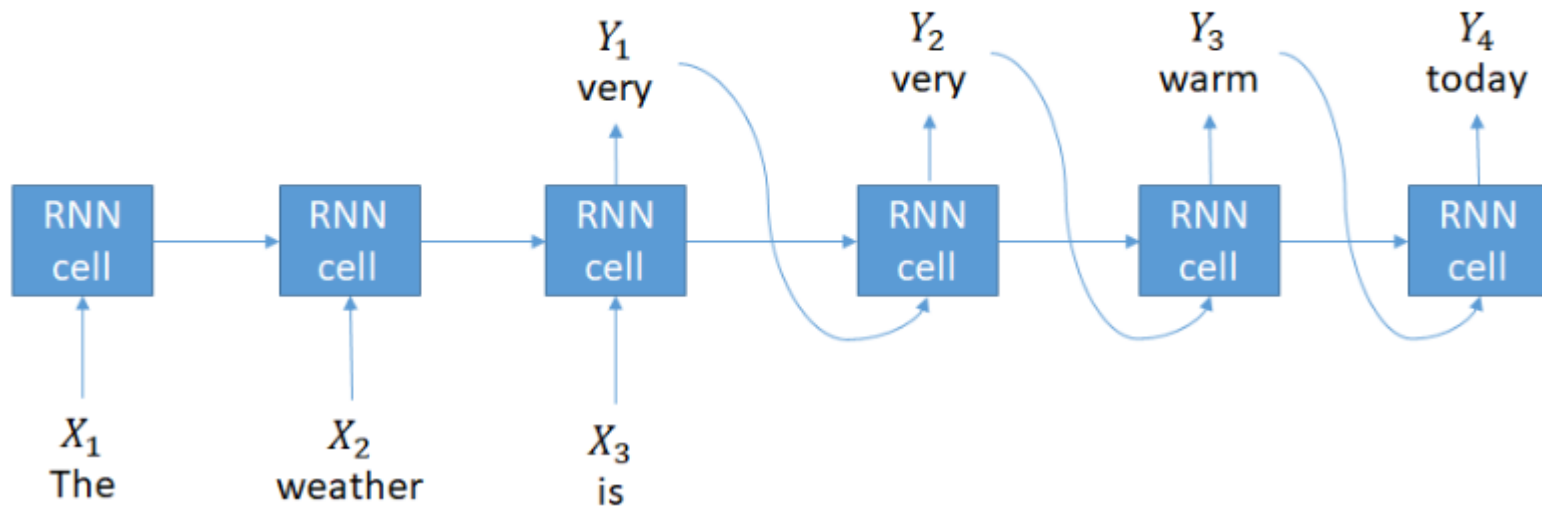
Decision rule:

Set prediction to “positive” if $Y_{pred} > 0.5$, otherwise set to “negative”.

* Y_{pred} is the probability of the sentence to belong to class “positive”

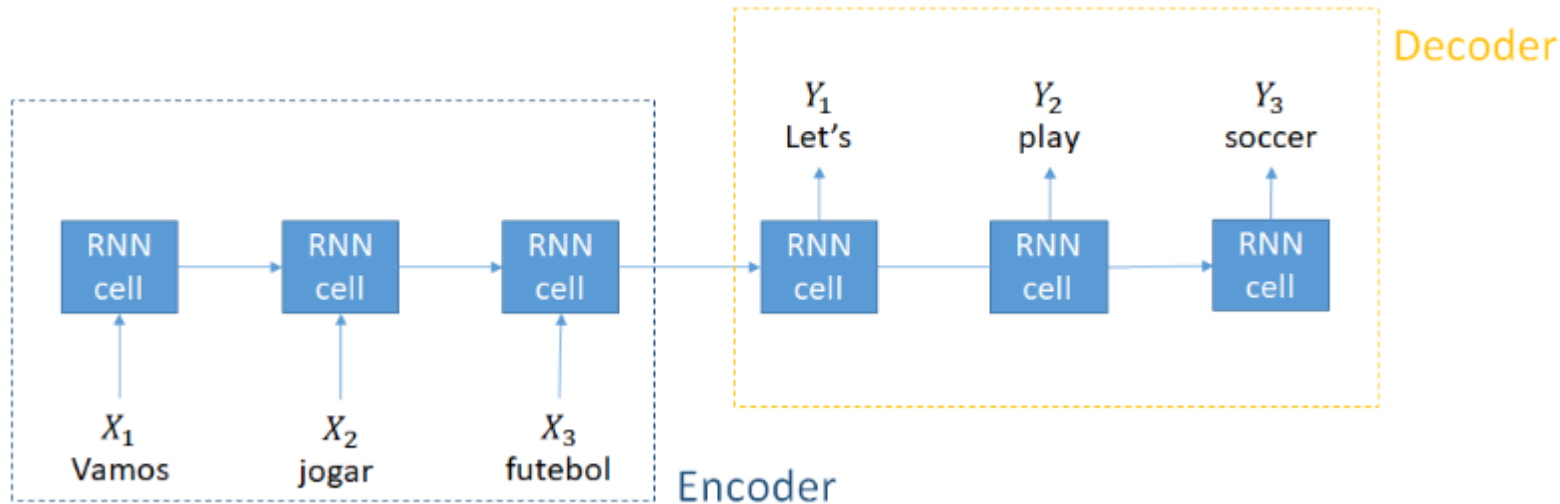
Sequence Models (2/3)

Many to many: text generation

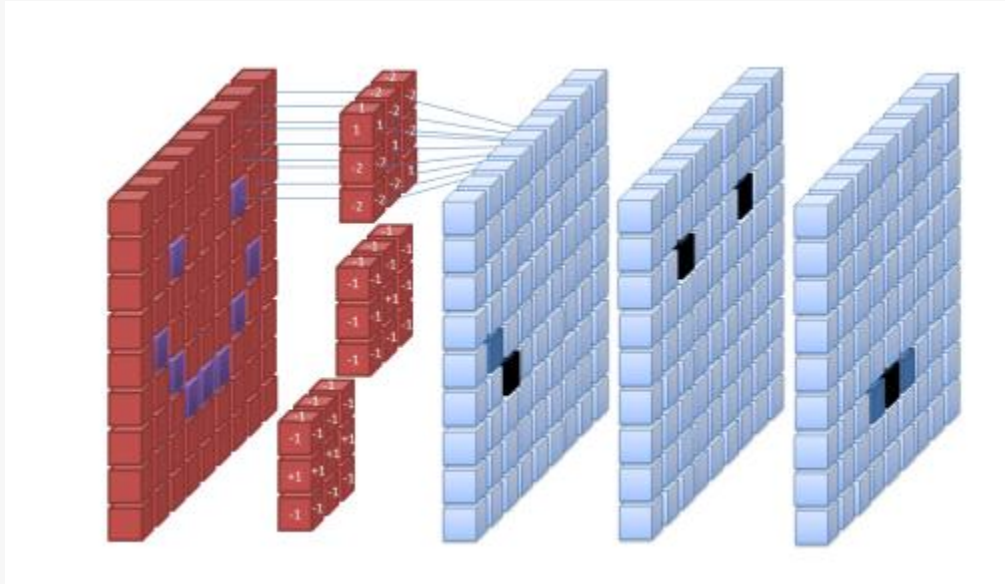


Sequence Models (3/3)

Many to many: neural machine translation



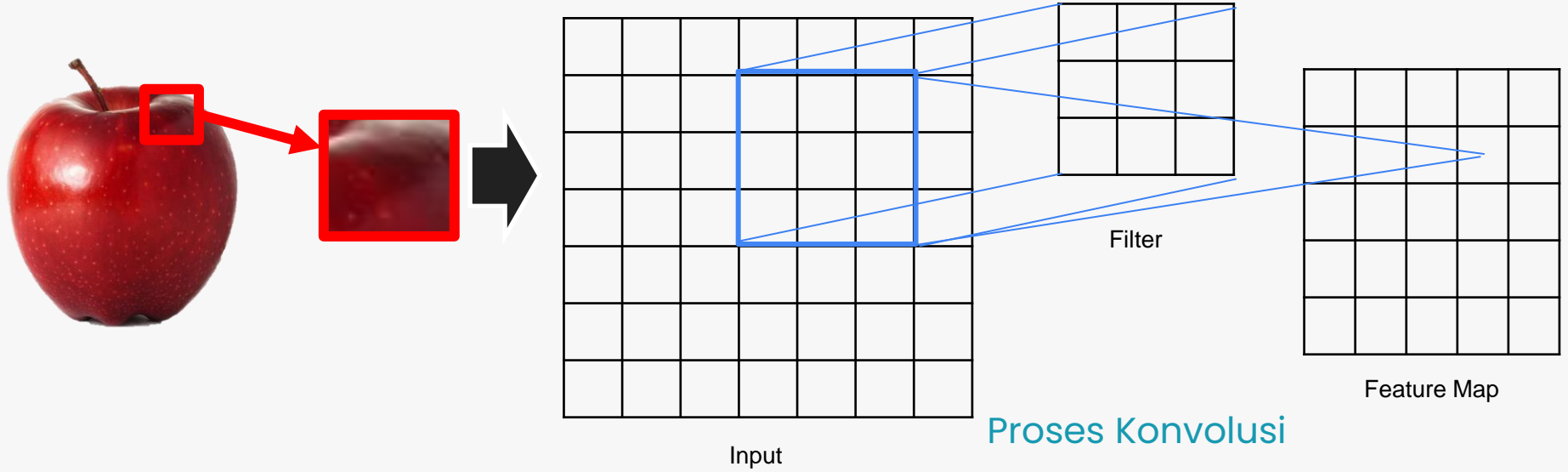
Convolutional



Penggunaan: Gambar dan Video

Convolutional Neural Network





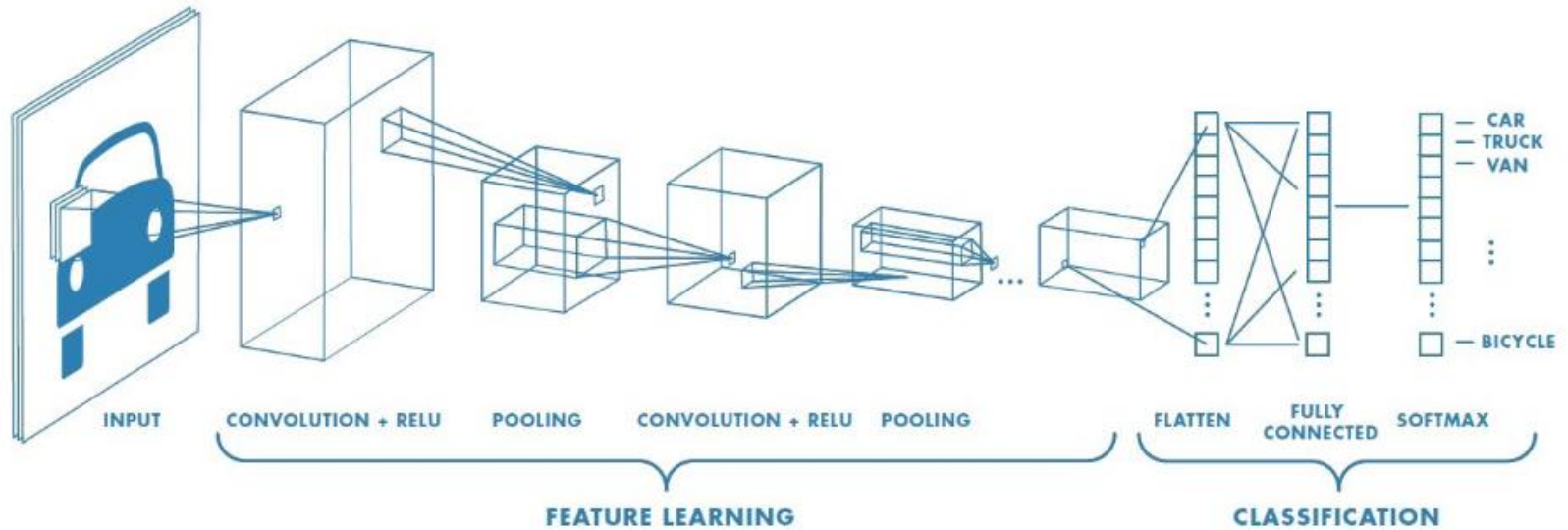
3_0	3_1	2_2	1	0
0_2	0_2	1_0	3	1
3_0	1_1	2_2	2	3
2	0	0	2	2
2	0	0	0	1

*

0	1	2
2	2	0
0	1	2



12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0



Activation functions

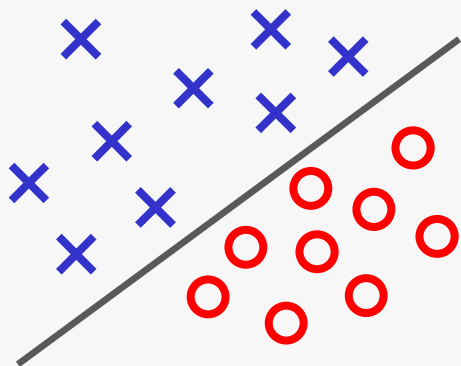
Activation Function

Membuat *neural network* menjadi non-linear

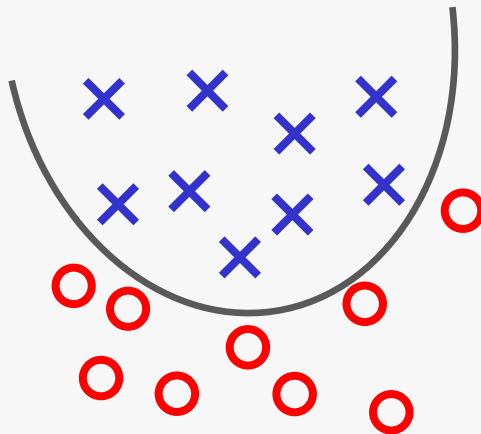


Linear vs. Non-linear

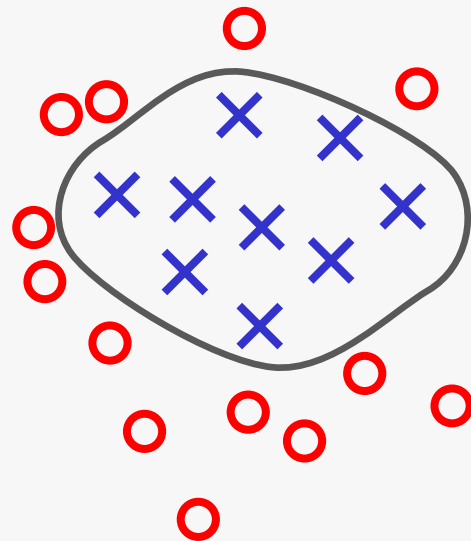
Linear



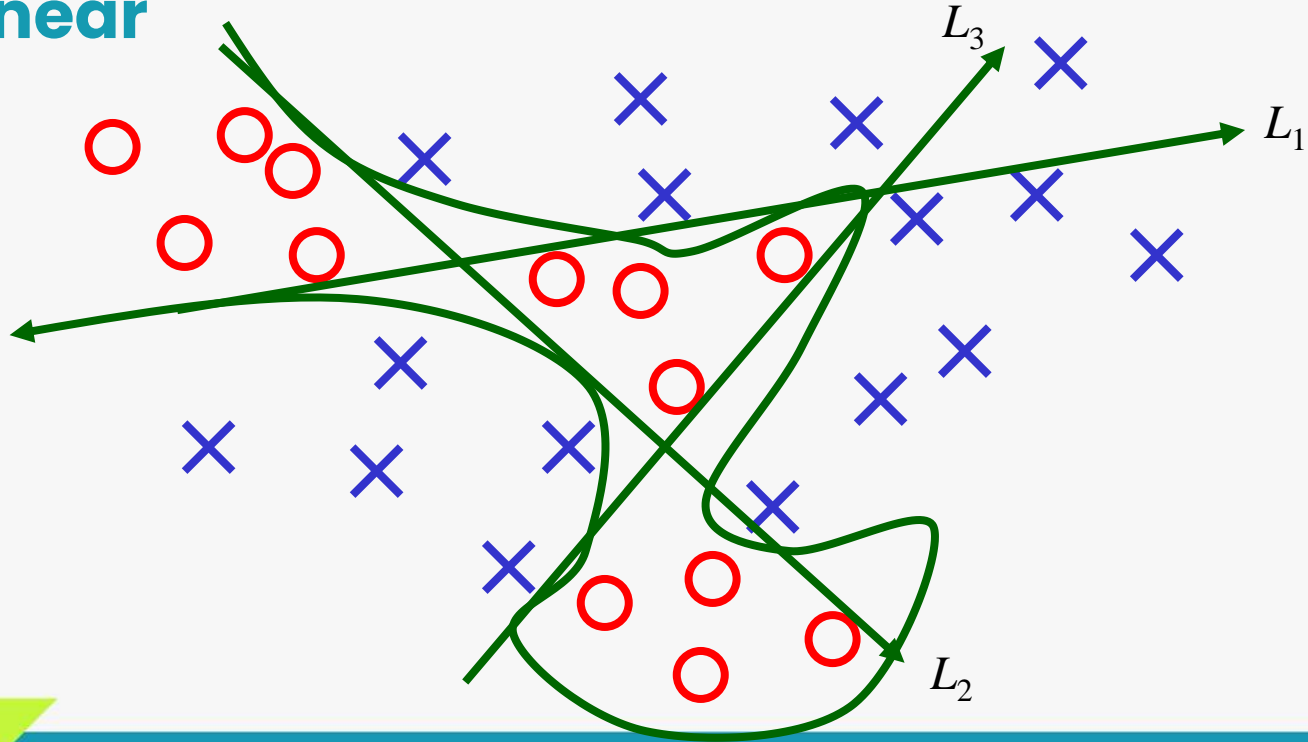
Linear



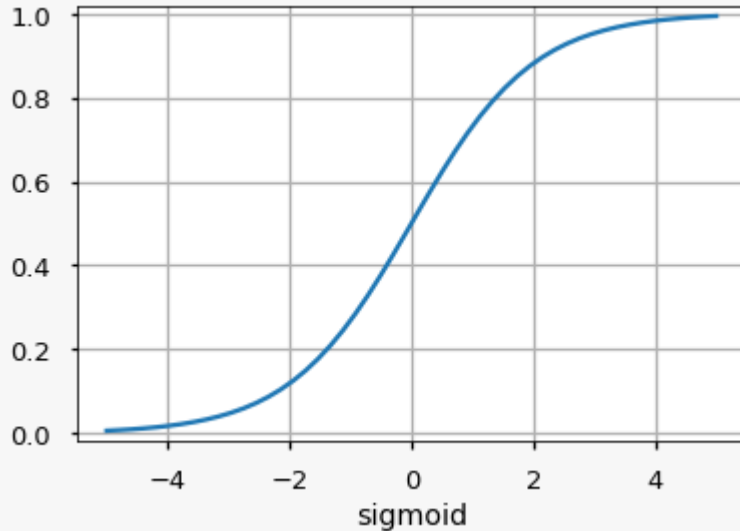
Non-linear



Permasalahan Non-linear



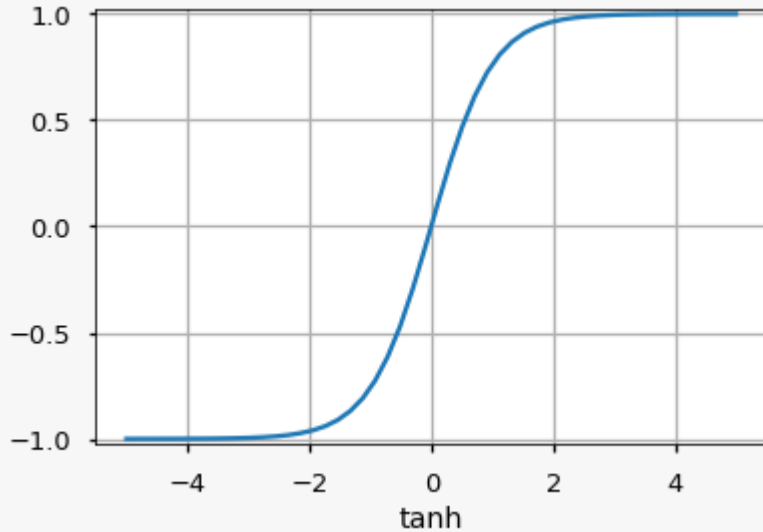
Sigmoid dan Softmax (1/3)



Menerima inputan dari nilai 0 sampai dengan 1 cocok untuk binary classification sedangkan softmax cocok untuk multiclass.

$$a(f) = \frac{1}{1 + \exp(-f)}$$

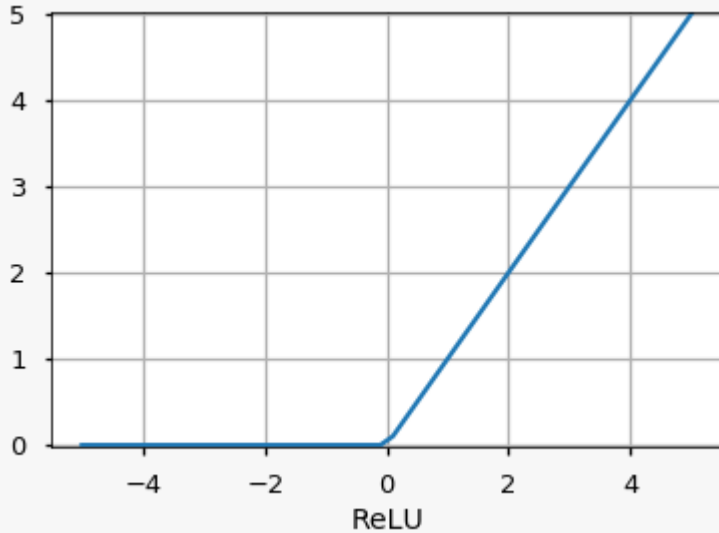
Tanh (2/3)



Menerima inputan dari nilai -1 sampai dengan 1

$$\tanh(x) = 2\sigma(2x) - 1$$

ReLU (3/3)

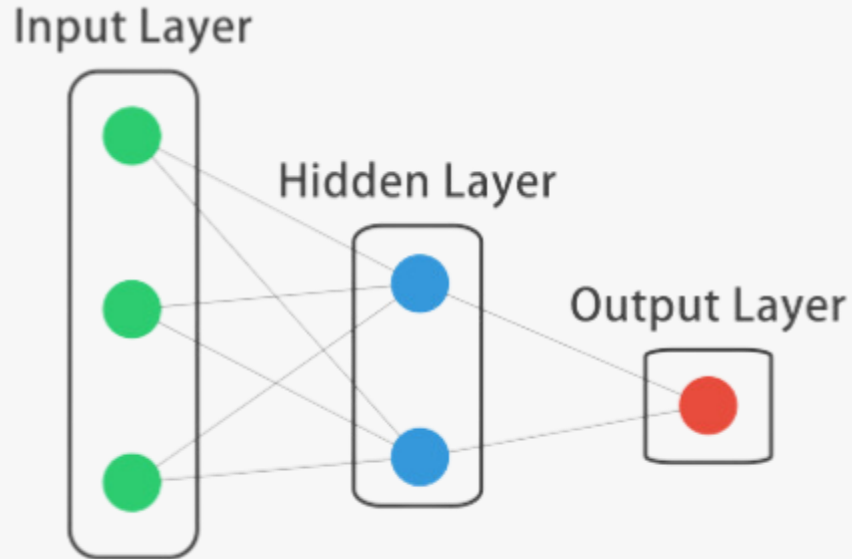


Menerima inputan dari nilai positif saja,
angka negatif ditulis 0

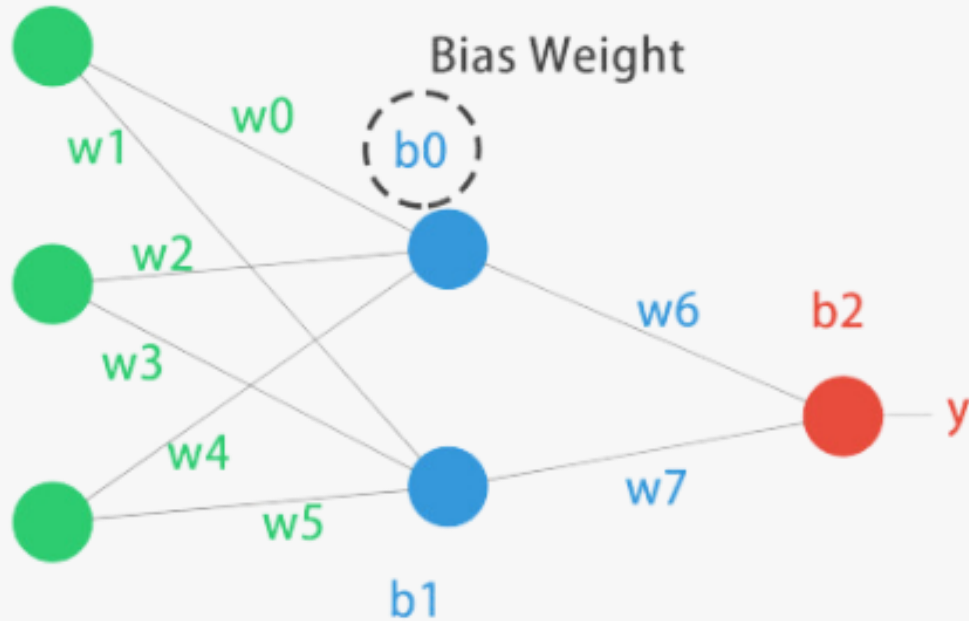
$$f(x) = \max(0, x)$$

Building a simple neural network using Keras

Inisiasi Neural Networks (1/2)



Inisiasi Neural Networks (2/2)



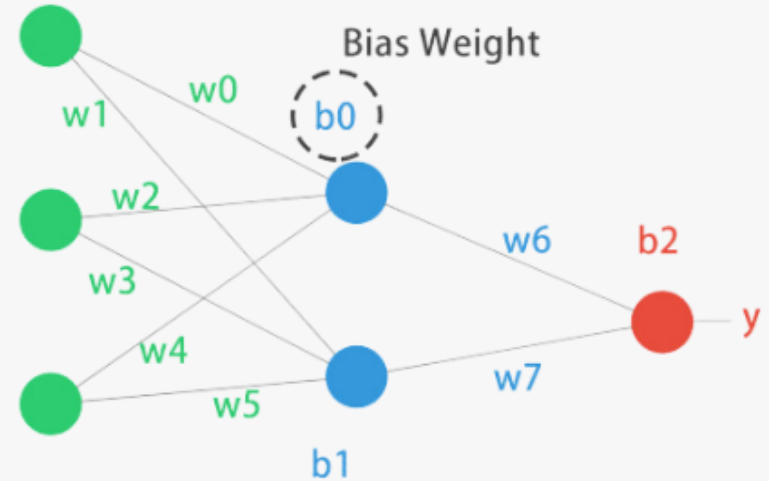
Membuat Neural Networks

```
from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense
```

```
# Membuat Sequential Model  
model = Sequential()
```

```
# Menambahkan dense layer  
model.add(Dense(2, input_shape=(3,)))
```

```
# Menambahkan dense layer output  
model.add(Dense(1))
```



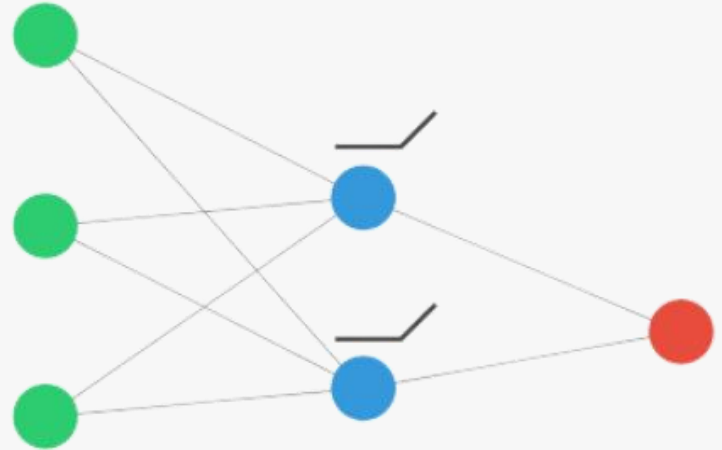
Menambahkan Activation Function

```
from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense
```

```
# Membuat Sequential Model  
model = Sequential()
```

```
# Menambahkan dense layer  
model.add(Dense(2, input_shape=(3,),  
               activation = "relu"))
```

```
# Menambahkan dense layer output  
model.add(Dense(1))
```

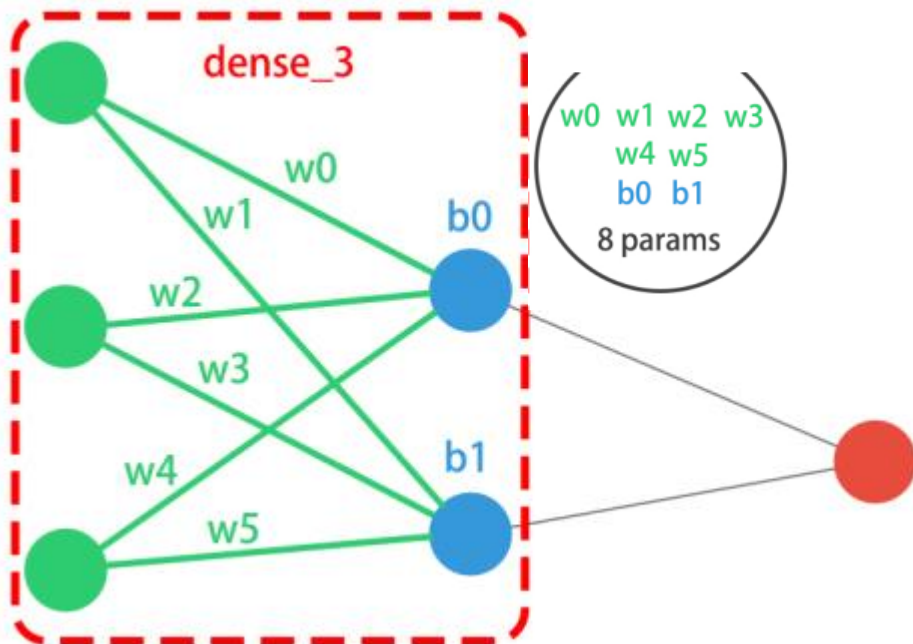


Model Summarize (1/2)

```
model.summary()
```

```
Layer (type)                 Output Shape              Param #
=====
dense_3 (Dense)              (None, 2)                 8
-----
dense_4 (Dense)              (None, 1)                 3
=====
Total params: 11
Trainable params: 11
Non-trainable params: 0
-----
```

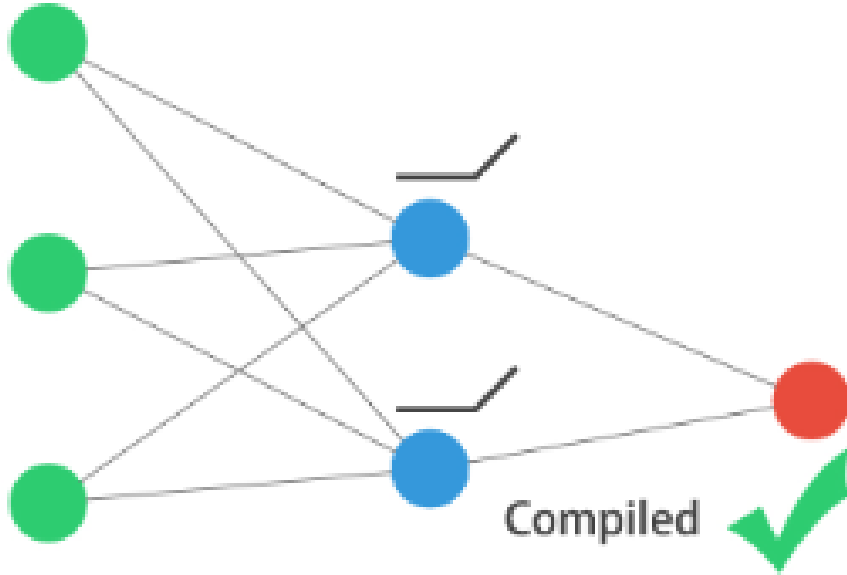
Model Summarize (2/2)



```
model.summary()
```

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 2)	--> 8 <--
dense_4 (Dense)	(None, 1)	3
Total params: 11		
Trainable params: 11		
Non-trainable params: 0		

Compiling



```
model.compile(optimizer="adam",  
              loss="mse")
```

- **Adam** adalah salah satu algoritma optimasi yang populer dan efisien dalam melakukan penyesuaian bobot
- **MSE** adalah salah satu fungsi loss untuk mengukur seberapa besar selisih antara prediksi model dengan nilai sebenarnya (ground truth).

Training

```
model.fit(X_train, y_train, epochs=5)
```

```
Epoch 1/5  
1000/1000 [=====] - 0s 242us/step - loss: 0.4090  
Epoch 2/5  
1000/1000 [=====] - 0s 34us/step - loss: 0.3602  
Epoch 3/5  
1000/1000 [=====] - 0s 37us/step - loss: 0.3223  
Epoch 4/5  
1000/1000 [=====] - 0s 34us/step - loss: 0.2958  
Epoch 5/5  
1000/1000 [=====] - 0s 33us/step - loss: 0.2795
```

Predict

```
model.predict(X_test)
```

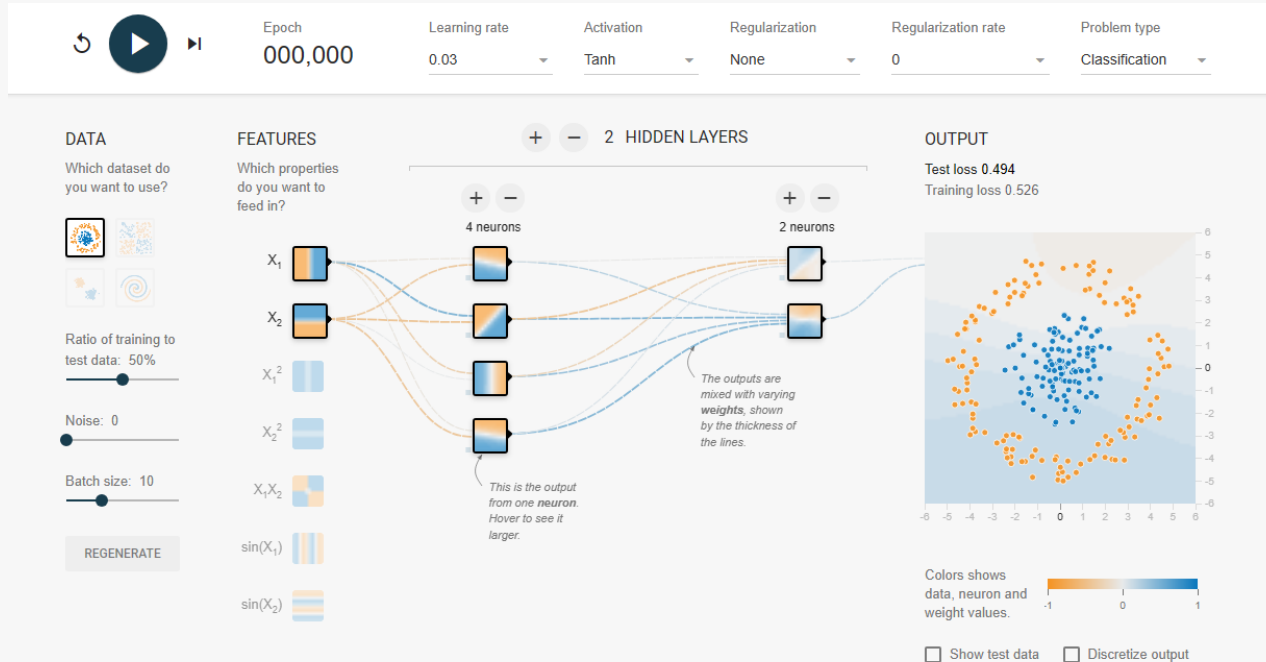
```
array([[0.6131608 ],  
       [0.5175948 ],  
       [0.60209155],  
       ...,  
       [0.55633    ],  
       [0.5305591 ],  
       [0.50682044]])
```

Evaluating

```
model.evaluate(X_test, y_test)
```

```
1000/1000 [=====] - 0s 53us/step  
0.25
```

Playground Neural Networks



Kita akan melakukan elaborasi pemahaman Neural Networks ([Klik](#))

Thanks!

