



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS BRAWIJAYA**

BAB : KONSEP JST  
NAMA : ARION SYEMAEL SIAHAAN  
NIM : 225150207111060  
TANGGAL : 04/09/2024  
ASISTEN : ALIFAH KHAIRUNNISA  
ANDHIKA IHSAN CENDEKIA



**A. Praktikum**

1. Buka Google Collaboratory melalui [tautan ini](#).
2. Tulis kode berikut ke dalam setiap *cell* pada *notebook* tersebut.

**a. Fungsi Step Biner**

```
def binstep(x, th=0):  
    return 1 if x >= th else 0
```

**b. Fungsi McCulloch-Pitts (MCP) Neuron**

```
import numpy as np  
  
def MCP(x, w, th):  
    y_in = np.dot(x, w)  
    y_out = binstep(y_in, th)  
    return y_out
```

**c. Fungsi Hitung Akurasi**

```
def calc_accuracy(a, b):  
    s = [1 if a[i] == b[i] else 0 for i in range(len(a))]  
    return sum(s) / len(a)
```

**d. Fungsi Logika AND**

```
def AND(X):  
    w = [1, 1]  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
    return y
```

**e. Eksekusi Logika AND**

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]  
output = AND(data)  
true = [0, 0, 0, 1]  
accuracy = calc_accuracy(output, true)  
print('Output:', output)  
print('True:', true)
```

```
print('Accuracy:', accuracy)
```

#### f. Fungsi Logika OR

```
def OR(X):  
    w = [2, 2]  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
    return y
```

#### g. Eksekusi Logika OR

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]  
output = OR(data)  
true = [0, 1, 1, 1]  
accuracy = calc_accuracy(output, true)  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

#### h. Logika AND NOT

```
def ANDNOT(X):  
    w = [2, -1]  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
    return y
```

#### i. Eksekusi Logika AND NOT

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]  
output = ANDNOT(data)  
true = [0, 0, 1, 0]  
accuracy = calc_accuracy(output, true)  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

#### j. Fungsi Logika XOR

```
def XOR(X):  
    X_flip = [(i[1], i[0]) for i in X]  
    y1 = ANDNOT(X)  
    y2 = ANDNOT(X_flip)  
    y = [OR([(y1[i], y2[i])]) for i in range(len(y1))]  
    return [yi[0] for yi in y]
```

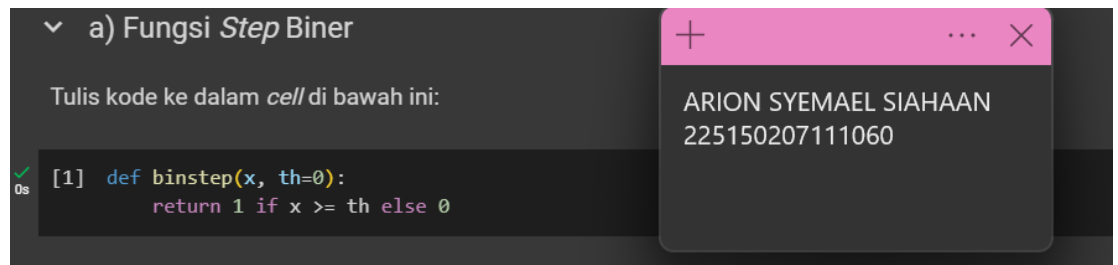
#### k. Eksekusi Logika XOR

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]
```

```
output = XOR(data)
true = [0, 1, 1, 0]
accuracy = calc_accuracy(output, true)
print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)
```

## B. Screenshot

### a. Fungsi *Step* Biner



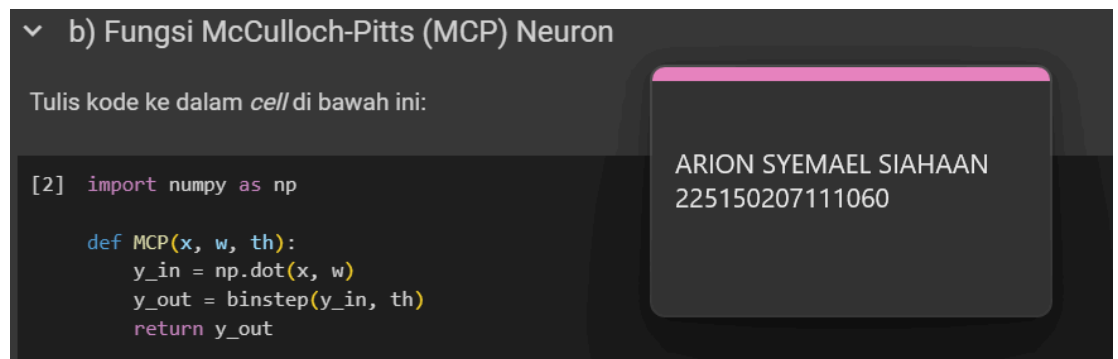
```
▼ a) Fungsi Step Biner
```

Tulis kode ke dalam *cell* di bawah ini:

```
[1] def binstep(x, th=0):
    return 1 if x >= th else 0
```

ARION SYEMAEL SIAHAAN  
225150207111060

### b. Fungsi McCulloch-Pitts (MCP) Neuron



```
▼ b) Fungsi McCulloch-Pitts (MCP) Neuron
```

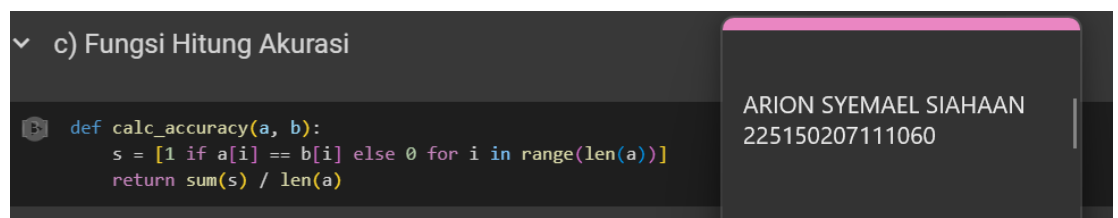
Tulis kode ke dalam *cell* di bawah ini:

```
[2] import numpy as np

def MCP(x, w, th):
    y_in = np.dot(x, w)
    y_out = binstep(y_in, th)
    return y_out
```

ARION SYEMAEL SIAHAAN  
225150207111060

### c. Fungsi Hitung Akurasi



```
▼ c) Fungsi Hitung Akurasi
```

```
def calc_accuracy(a, b):
    s = [1 if a[i] == b[i] else 0 for i in range(len(a))]
    return sum(s) / len(a)
```

ARION SYEMAEL SIAHAAN  
225150207111060

### d. Fungsi Logika AND

#### ▼ d) Fungsi Logika AND

Tulis kode ke dalam *cell* di bawah ini:

```
def AND(X):  
    w = [1, 1]  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
    return y
```

ARION SYEMAEL SIAHAAN  
225150207111060

#### e. Eksekusi Logika AND

#### ▼ e) Eksekusi Logika AND

Tulis kode ke dalam *cell* di bawah ini:

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]  
output = AND(data)  
true = [0, 0, 0, 1]  
accuracy = calc_accuracy(output, true)  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

```
➦ Output: [0, 0, 0, 1]  
True: [0, 0, 0, 1]  
Accuracy: 1.0
```

ARION SYEMAEL SIAHAAN  
225150207111060

#### f. Fungsi Logika OR

#### ▼ f) Fungsi Logika OR

Tulis kode ke dalam *cell* di bawah ini:

```
def OR(X):  
    w = [2, 2]  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
    return y
```

ARION SYEMAEL SIAHAAN  
225150207111060

g. Eksekusi Logika OR

g) Eksekusi Logika OR

Tulis kode ke dalam *cell* di bawah ini:

```
[7] data = [(0, 0), (0, 1), (1, 0), (1, 1)]
      output = OR(data)
      true = [0, 1, 1, 1]
      accuracy = calc_accuracy(output, true)
      print('Output:', output)
      print('True:', true)
      print('Accuracy:', accuracy)
```

Output: [0, 1, 1, 1]  
True: [0, 1, 1, 1]  
Accuracy: 1.0

ARION SYEMAEL SIAHAAN  
225150207111060

h. Logika AND NOT

h) Logika AND NOT

Tulis kode ke dalam *cell* di bawah ini:

```
[8] def ANDNOT(X):
      w = [2, -1]
      th = 2
      y = [MCP(i, w, th) for i in X]
      return y
```

ARION SYEMAEL SIAHAAN  
225150207111060

i. Eksekusi Logika AND NOT

i) Eksekusi Logika AND NOT

Tulis kode ke dalam *cell* di bawah ini:

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]
output = ANDNOT(data)
true = [0, 0, 1, 0]
accuracy = calc_accuracy(output, true)
print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)
```

Output: [0, 0, 1, 0]  
True: [0, 0, 1, 0]  
Accuracy: 1.0

ARION SYEMAEL SIAHAAN  
225150207111060

j. Logika XOR

▼ j) Fungsi Logika XOR

Tulis kode ke dalam *cell* di bawah ini:

```
[10] def XOR(X):  
    X_flip = [(i[1], i[0]) for i in X]  
    y1 = ANDNOT(X)  
    y2 = ANDNOT(X_flip)  
    y = [OR((y1[i], y2[i])) for i in range(len(y1))]  
    return [yi[0] for yi in y]
```

ARION SYEMAEL SIAHAAN  
225150207111060

k. Eksekusi Logika XOR

▼ k) Eksekusi Logika XOR

Tulis kode ke dalam *cell* di bawah ini:

```
data = [(0, 0), (0, 1), (1, 0), (1, 1)]  
output = XOR(data)  
true = [0, 1, 1, 0]  
accuracy = calc_accuracy(output, true)  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

Output: [0, 1, 1, 0]  
True: [0, 1, 1, 0]  
Accuracy: 1.0

ARION SYEMAEL SIAHAAN  
225150207111060

### C. Analisis

1. Jalankan semua cell (Ctrl+F9). Amati output dan akurasi yang dihasilkan.
2. Pada kode a, apa maksud dari statement `return 1 if x >= th else 0`?  
Ini adalah fungsi aktivasi step biner yang menghasilkan output 1 jika input x lebih besar atau sama dengan ambang batas th, dan 0 jika lebih kecil. Digunakan untuk mensimulasikan keputusan biner dalam neuron tiruan.
3. Pada kode b:
  - a. Apakah perbedaan antara variabel `y_in` dan `y_out`?  
`y_in` adalah nilai total input yang dihitung dari hasil perkalian bobot w dan input x (menggunakan dot product). `y_out` adalah hasil dari fungsi aktivasi binstep yang mengubah `y_in` menjadi output biner (0 atau 1).

b. Apakah yang dilakukan oleh fungsi `np.dot()`?

`np.dot()` menghitung hasil perkalian dot (dot product) antara dua vektor (input  $x$  dan bobot  $w$ ). Ini menggabungkan input dengan bobot untuk menghitung total input ke neuron.

4. Pada kode B, apakah kegunaan dari variabel-variabel  $w$  dan  $th$ ?

Pada kode b, variabel  $w$  adalah bobot yang menunjukkan seberapa besar kontribusi masing-masing input terhadap keputusan akhir neuron. Setiap input yang diterima neuron akan dikalikan dengan bobot terkait. Sedangkan  $th$  (threshold) adalah ambang batas yang menentukan apakah output neuron adalah 1 atau 0, tergantung pada total input yang diterima neuron setelah proses agregasi input dengan bobot. Jika total input melebihi atau sama dengan threshold, outputnya 1; jika tidak, outputnya 0.

5. Pada kode j, apakah tujuan dari variabel  $X\_flip$ ?

$X\_flip$  membalik urutan input dalam pasangan (tuple). Hal ini digunakan untuk mengimplementasikan logika ANDNOT secara simetris pada kedua input dalam operasi XOR.

6. Pada hal apa saja terdapat kesamaan antara neuron biologis dan neuron tiruan?

Keduanya menerima input, memiliki bobot (analog dengan sinapsis pada neuron biologis), menerapkan fungsi aktivasi, dan menghasilkan output biner. Namun, neuron biologis lebih kompleks dengan sinyal non-biner dan berbagai pola interaksi.

7. Jelaskan kelebihan dan kekurangan yang dimiliki McCulloch-Pitts neuron.

McCulloch-Pitts neuron memiliki beberapa kelebihan, di antaranya adalah kesederhanaannya, yang membuatnya mudah dipahami dan diimplementasikan. Model ini mampu merepresentasikan logika biner dasar seperti AND, OR, dan NOT, sehingga cocok untuk aplikasi sederhana yang memerlukan keputusan biner. Namun, neuron McCulloch-Pitts juga memiliki kekurangan yang signifikan. Model ini tidak mampu menangani masalah yang lebih kompleks, seperti operasi XOR, tanpa menggunakan kombinasi beberapa neuron. Selain itu, fungsi aktivasi biner yang digunakan terlalu terbatas untuk aplikasi praktis yang memerlukan analisis lebih rumit, seperti pengenalan pola atau klasifikasi data yang tidak bersifat linear. Keterbatasan ini membuatnya kurang efektif dalam banyak aplikasi modern kecerdasan buatan.

8. Mengapa logika XOR lebih rumit sehingga membutuhkan kombinasi beberapa logika yang lain?

XOR memerlukan keluaran 1 hanya ketika satu input benar dan input lain salah. Ini tidak bisa direpresentasikan dengan kombinasi linear sederhana seperti AND atau

OR, sehingga perlu kombinasi dari beberapa fungsi logika dasar (ANDNOT) untuk menangani kasus asimetris tersebut.

#### **D. Kesimpulan**

1. Jaringan Syaraf Tiruan (JST) adalah model komputasi yang terinspirasi oleh cara kerja otak manusia. JST terdiri dari sejumlah unit sederhana yang disebut neuron, yang dihubungkan satu sama lain untuk membentuk jaringan. Neuron-neuron ini bekerja dengan cara menerima input, memprosesnya dengan bobot tertentu, lalu menghasilkan output melalui fungsi aktivasi. JST digunakan untuk memecahkan masalah seperti pengenalan pola, klasifikasi, dan prediksi.
2. Dalam Jaringan Syaraf Tiruan (JST),  $x$  merupakan input yang diterima oleh neuron. Bobot atau  $w$  mengalikan setiap input untuk menentukan seberapa besar pengaruh input tersebut terhadap output neuron. Output neuron,  $y$ , dihasilkan setelah input dikalikan dengan bobot dan ditambahkan bias. Fungsi aktivasi adalah fungsi matematika yang diterapkan pada hasil agregasi input tersebut, yang berperan penting dalam menentukan output akhir neuron serta memperkenalkan non-linearitas ke dalam model jaringan.
3. McCulloch-Pitts neuron adalah model neuron tiruan sederhana yang terdiri dari beberapa input biner, bobot, fungsi aktivasi biner (step function), dan satu output biner. Input yang diterima dikalikan dengan bobot, kemudian dijumlahkan untuk menghasilkan nilai total input. Jika nilai total tersebut melebihi ambang batas (threshold), neuron menghasilkan output **1**, dan jika tidak, outputnya adalah **0**. Model ini digunakan untuk merepresentasikan fungsi logika dasar seperti AND, OR, dan NOT, namun terbatas dalam menangani logika yang lebih kompleks seperti XOR.