Tugas Besar A
IF3270 Pembelajaran Mesin

Implementasi *Forward Propagation* untuk *Feed Forward Neural Network*

Disusun oleh:

13519096 Girvin Junod

13519116 Jeane Mikha Erwansyah

13519131 Hera Shafira

13519188 Jeremia Axel Bachtera

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Semester 2 Tahun 2021/2022

1. Penjelasan Program

Program ini adalah implementasi *forward propagation* untuk *Feedforward Neural Network* (FFNN). Model untuk FFNN disimpan dalam file JSON dan dapat di-*load* ke program. Dalam program ini, digunakan *compact model* untuk merepresentasikan FFNN di dalam program. Ini berarti *layer* yang berisi neuron direpresentasikan dengan matriks. Ini dapat dilihat juga dalam model yang disimpan dalam file JSON. Ini dilakukan untuk memudahkan pemrosesan data dari file eksternal ke program dan dalam manipulasi FFNN di dalam program.

Dalam implementasi program di bahasa Python ini, dibuat dua *class* yaitu *class* `Layer` dan `Graph` sebagai model dari FFNN di dalam program. *Class* Layer adalah *class* representasi dari *layer* pada FFNN. *Class* ini memiliki atribut berupa `weights` yaitu *weight dari* neuron-neuron yang ada di *layer*, `bias_weights` yang merupakan *weight* dari bias yang ada di *layer*, `values` yang merupakan *value-value* yang ada di neuron yang ada di *layer*, `label` yang merupakan label dari neuron untuk kebutuhan visualisasi, `type` yang menandakan tipe dari *layer*, dan `activation_func` yang menandakan fungsi aktivasi dari *layer*.

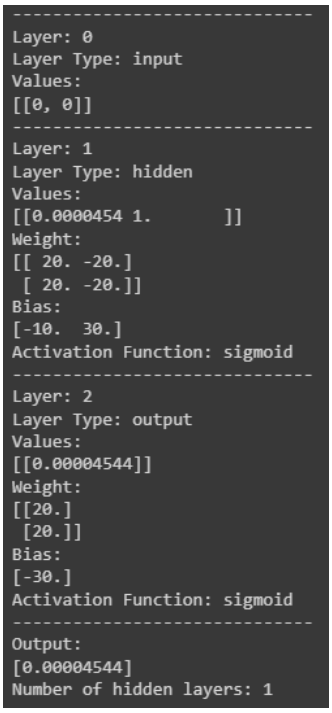
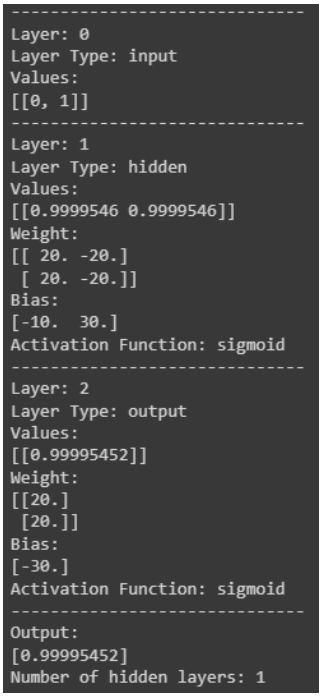
Class `Graph` merupakan representasi dari kumpulan-kumpulan *layer* yang membentuk FFNN. *Class* ini mempunyai satu atribut saja yaitu `layers` yang merupakan *list* layer-layer yang ada di FFNN. Pada *class* ini, dibuat beberapa *method* terkait dengan FFNN seperti *method* untuk membuat *layer* kosong, membuat *input layer*, menambah *layer*, menghitung *value* dari *layer*, menghitung *net value* dari *input*, *method-method* untuk fungsi aktivasi, untuk prediksi *output*, untuk me-*load* graf dari file eksternal, dan untuk memvisualisasi model.

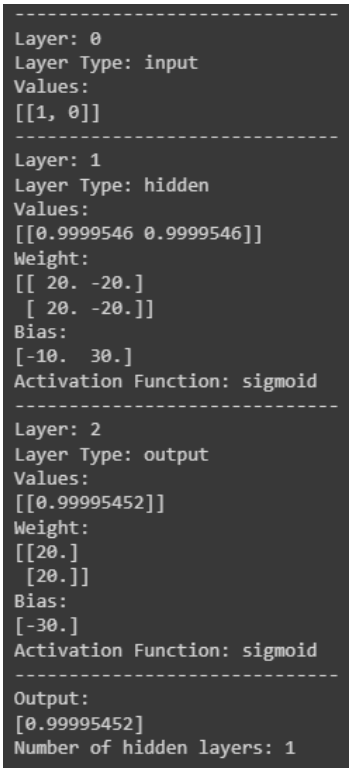
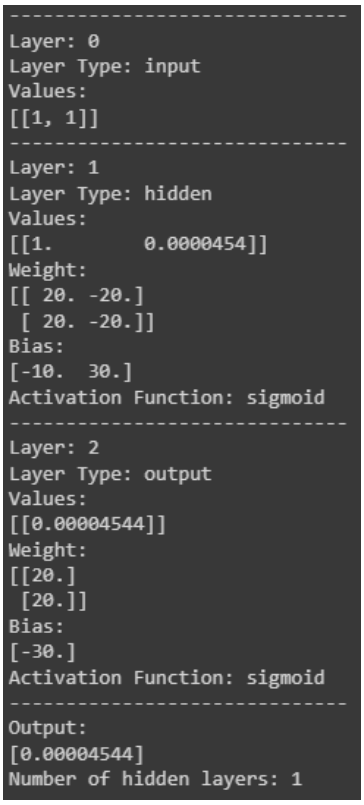
Untuk melakukan *predict* output dari suatu input, pertama dilakukan *load* model dari FFNN yang disimpan di file eksternal. Lalu, dilakukan operasi berupa perkalian matriks *weights* dengan matriks *value* pada *layer* dan ditambah matriks bias pada tiap *layer*-nya. Terdapat *method* predict pada kelas `Graph` untuk menghasilkan output dari input yang diberikan melalui parameter *method*.

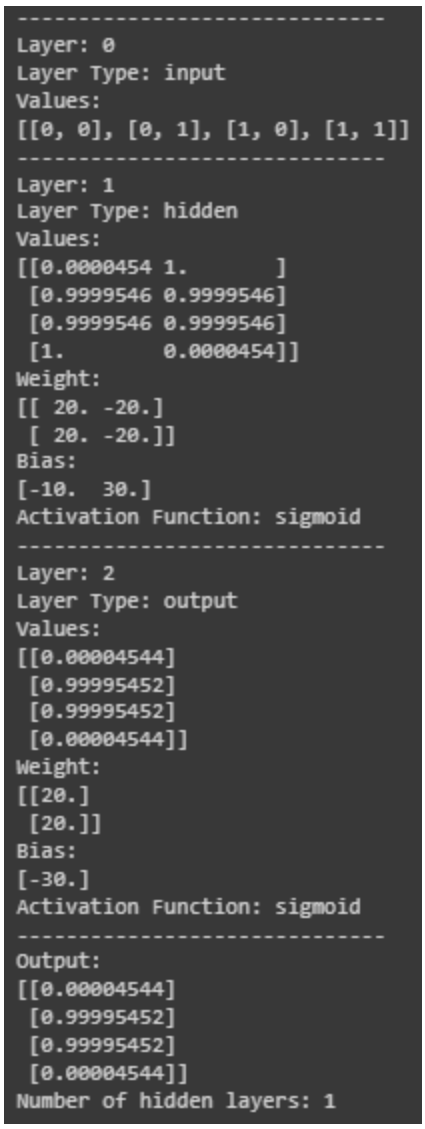
Secara keseluruhan, langkah untuk menjalankan program ini adalah dengan pertama, memanggil konstruktor kelas `Graph`. Kemudian melakukan *load* model dari file dengan memanggil *method* `load_graph` dari kelas `Graph`. Kemudian, panggil *method* `predict` dengan parameter input yang ingin dimasukkan ke dalam model, *method* ini akan mengembalikkan matriks output dari model. *Print* model dan output dengan memanggil *method* `display_table`.

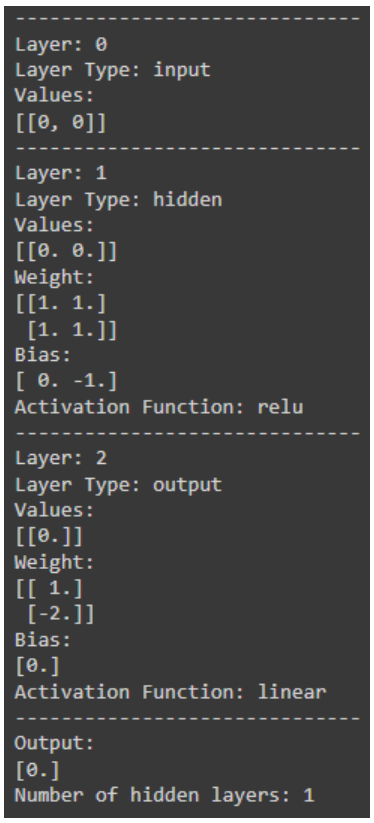
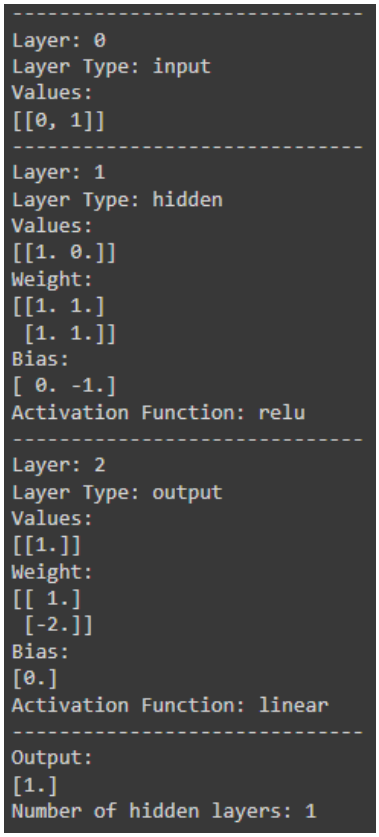
2. Hasil Pengujian

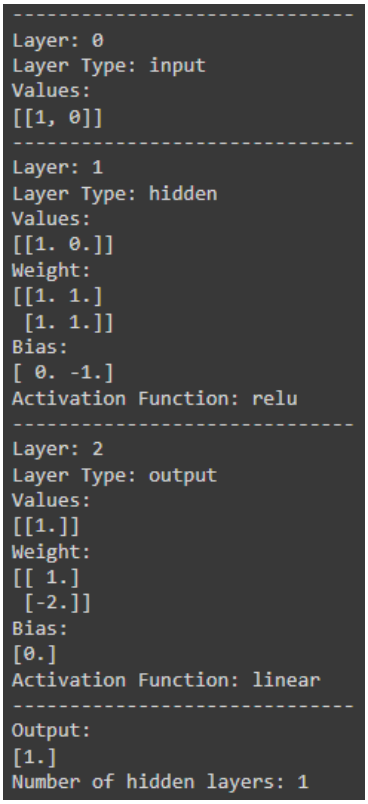
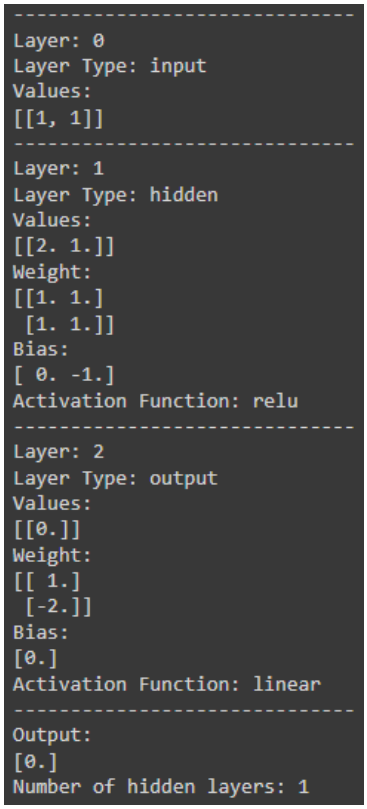
Pengujian dilakukan dengan input sesuai pada slide XOR sigmoid dan XOR ReLU + Linear pada PPT kuliah.

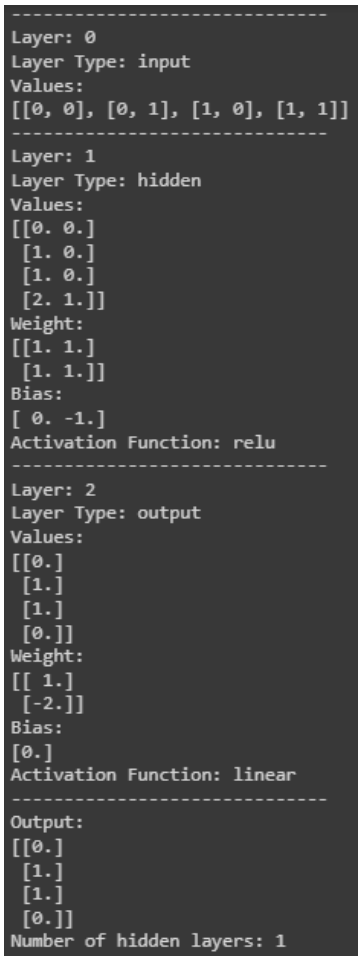
No.	Jenis Input	Input	Fungsi Aktivasi	Screenshot
1.	1 instance	[0,0]	Sigmoid	
2.	1 instance	[0,1]	Sigmoid	

3.	1 instance	[1,0]	Sigmoid	
4.	1 instance	[1,1]	Sigmoid	

5.	Batch	[[0,0],[0,1], [1,0],[1,1]]	Sigmoid	

6.	1 instance	[0,0]	ReLU +Linear	
7.	1 instance	[0,1]	ReLU +Linear	

8.	1 instance	[1,0]	ReLU +Linear	
9.	1 instance	[1,1]	ReLU +Linear	

10.	Batch	[[0,0],[0,1], [1,0],[1,1]]	ReLU +Linear	

3. Perbandingan Hasil Dengan Perhitungan Manual

Berikut adalah perbandingan hasil program dengan perhitungan manual yang terdapat pada PPT kuliah.

<table>
<tr><th>No.</th><th>Fungsi Aktivasi</th><th>Hasil Program</th><th>Perhitungan Manual</th></tr>
<tr><td>1.</td><td>Sigmoid</td><td>

```
Output:
[[0.00004544]
 [0.99995452]
 [0.99995452]
 [0.00004544]]
Number of hidden layers: 1
```

</td><td>

x0	x1	x2	f	∑h1	h1	∑h2	h2	∑y	y
1	0	0	0	-10	0.00	30	1.00	-10.00	0.00
1	0	1	1	10	1.00	10	1.00	10.00	1.00
1	1	0	1	10	1.00	10	1.00	10.00	1.00
1	1	1	0	30	1.00	-10	0.00	-10.00	0.00

</td></tr>
<tr><td>2.</td><td>ReLU + Linear</td><td>

```
Output:
[[0.]
 [1.]
 [1.]
 [0.]]
Number of hidden layers: 1
```

</td><td>

$w_{hy} = \begin{bmatrix} 1 \\ -2 \end{bmatrix} \Rightarrow y = hw_{hy} + b = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$

$b = 0$

</td></tr>
</table>

Pada output fungsi aktivasi Sigmoid, dapat dilihat terdapat sedikit perbedaan antara hasil program dengan perhitungan manual. Hal ini terjadi karena pada perhitungan manual dilakukan pembulatan. Apabila dilakukan pembulatan pada hasil program fungsi aktivasi Sigmoid maka akan diperoleh [0,1,1,0] yang sama dengan hasil perhitungan manual. Pada output fungsi aktivasil ReLU + Linear, dapat dilihat bahwa hasil program sudah sesuai dengan hasil perhitungan manual.

4. Pembagian Tugas

No.	NIM	Nama	Kontribusi
1	13519096	Girvin Junod	Model graf, fungsi aktivasi linear, fungsi aktivasi sigmoid, prediksi output
2	13519116	Jeane Mikha Erwansyah	Model graf, visualisasi graf, prediksi output
3	13519131	Hera Shafira	Model graf, fungsi aktivasi ReLU, fungsi aktivasi softmax, prediksi output
4	13519188	Jeremia Axel	Model graf, input file, prediksi

			output