IMPLEMENTASI *NEURAL NETWORK* UNTUK BANYAK RANGKAIAN KOMBINASIONAL

SKRIPSI

Oleh :

AKBAR HIDAYAT
2015031038

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

IMPLEMENTASI *NEURAL NETWORK* UNTUK BANYAK RANGKAIAN KOMBINASIONAL

Oleh

AKBAR HIDAYAT

Pengujian *Integrated Circuit* (IC) umumnya menggunakan *Generator Set Te*st (GST) untuk memastikan kualitas produksi. Pada setiap pengujian rangkaian memerlukan satu unit GST. Sehingga mengakibatkan banyak GST pada setiap IC pengujian. Hal ini membutuhkan biaya pengujian lebih besar. Prinsip kerja GST pada dasarnya adalah seperti suatu fungsi rangkaian digital kombinasional. Penelitian ini bertujuan untuk mengimplementasikan struktur *neural network* yang dapat menggantikan berbagai fungsi rangkaian kombinasional dengan syarat jumlah *input* dan *output* adalah sama. Sistem yang dirancang menggunakan lima fungsi rangkaian kombinasional, masing-masing dengan tiga *input* dan satu *output*, yang dimodelkan dalam arsitektur *neural network*. Karena struktur *input* dan *output* adalah sama, kelima rangkaian ini dapat digantikan dengan satu *neural network*, perbedaannya pada pengaturan bobotnya. Perbedaan bobot ini memungkinkan pemilihan rangkaian yang diaktifkan secara fleksibel. Peneliti ini adalah membuat algoritma yang mengatur penempatan bobot pada *neural network* sesuai dengan fungsi rangkaian yang diaktifkan. Hasil penelitian menunjukkan bahwa arsitektur yang diusulkan berhasil memodelkan kelima rangkaian kombinasional, memungkinkan penggunaan satu *neural network* yang fleksibel. Algoritma tersebut berhasil menyesuaikan bobot *neural network* untuk mencocokkan fungsi rangkaian yang diaktifkan, sehingga mengurangi biaya pengujian IC digital secara signifikan.

Kata Kunci: ***Neural Network,*** **Bobot, Kombinasional**

ABSTRACT

IMPLEMENTATION OF NEURAL NETWORKS FOR MANY COMBINATIONAL

By

AKBAR HIDAYAT

Integrated Circuit (IC) testing generally uses Generator Set Test (GST) to ensure production quality. Each network test requires one unit of GST. This results in a lot of GSTon each test IC. This requires greater testing costs. The working principle of GST is basically like a function of the digital network of combinationals. This research aims to implement a neural network structure that can replace various functions of the combinational circuit provided that the number of inputs and outputs is the same. The system is designed using five combinatorial circuit functions, each with three inputs and one output, modeled in a neural network architecture. Since the input and output structures are the same, these five circuits can be replaced by a single neural network, the difference being in the weight setting. This difference in weight allows for flexible selection of enabled circuits. This researcher is creating an algorithm that regulates the placement of weights on neural networks according to the function of the activated circuit. The results show that the proposed architecture successfully models all five sets of combinationals, allowing the use of a single flexible neural network. The algorithm successfully adjusts the neural network weights to match the functions of the enabled circuits, thereby significantly reducing the cost of testing digital ICs.

Keywords: Neural Network, Weight, Combinational

IMPLEMENTASI *NEURAL NETWORK* UNTUK BANYAK RANGKAIAN KOMBINASIONAL

Oleh :

AKBAR HIDAYAT

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar

SARJANA TEKNIK

Pada

Jurusan Teknik Elektro

Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2025

Judul Skripsi	: **IMPLEMENTASI *NEURAL NETWORK* UNTUK BANYAK RANGKAIAN KOMBINASIOANAL**
Nama Mahasiswa	: **Akbar Hidayat**
Nomor Pokok Mahasiswa	: 2015031038
Jurusan	: Teknik Elektro
Fakultas	: Teknik

MENYETUJUI

1. Komisi Pembimbing

Dr.Eng.Ageng Sadnowo R., S.T., M.T.	**Herlinawati, S.T., M.T.**
NIP. 19690228 199803 1 001	NIP. 19710314 199903 2 001

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Elektro
Herlinawati, S.T., M.T.	**Sumadi, S.T., M.T.**
NIP. 19710314 199903 2 001	NIP. 19731104 200003 1 001

MENGESAHKAN

1. Tim Penguji

Ketua : **Dr.Eng.Ageng Sadnowo R., S.T., M.T.** .....

Sekretaris : **Herlinawati, S.T., M.T.** .....

Penguji

Bukan Pembimbing : **Syaiful Alam, S.T., M.T.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : 24 Januari 2025

SURAT PERNYATAAN

Dengan ini saya menyatakan bahwa skripsi ini tidak terdapat karya yang pernah dilakukan orang lain dan sepanjang pengetahuan saya tidak terdapat atau diterbitkan oleh orang lain, kecuali secara tertulis diacu dalam naskah ini sebagaimana yang disebutkan dalam daftar pustaka. Selain itu, saya menyatakan pula bahwa skripsi saya ini dibuat oleh saya sendiri.

Apabila pernyataan saya tidak benar, maka saya bersedia dikenai sanksi akademik sesuai dengan hukum yang berlaku.

Bandar Lampung , 18 Februari 2025

Akbar Hidayat
NPM 2015031038

RIWAYAT HIDUP

Penulis dilahirkan di Lampung pada tanggal 22 Februari 2002, sebagai anak pertama dari dua bersaudara, dari pasangan Bapak Sapto Bugiono dan Ibu Nurhidayati. Riwayat pendidikan penulis dimulai dari SDS Gunung Bundar 3 pada tahun 2008 hingga 2014. Penulis melanjutkan tingkat sekolah menengah pertama di SMP N 1 Punduh Pedada pada tahun 2014 hingga 2017, kemudian penulis melanjutkan tingkat sekolah menengah atas di SMA N 1 Punduh Pedada pada tahun 2017-2020.

Penulis melanjutkan pendidikan ke jenjang perguruan tinggi di Universitas Lampung pada Program Studi Teknik Elektro tahun 2020 melalui jalur Ujian Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN). Selama menimbah ilmu di Jurusan Teknik Elektro, saya mempelajari komponen komponen elektronika serta dasar-dasar eloktronika. Pada tahun 2021 hingga 2022 saya menjadi bagian dari himpunan mahasiswa elektro sebagai anggota pada divisi sosial dan divisi minat dan bakat. Penulis melaksanakan Magang di PT. PLN (Persero) Unit Induk Distribusi Lampung selama empat bulan pada tahun 2023 pada Bidang Perencanaan Sistem Kelistrikan serta mengambil judul laporan yang berjudul "Perencanaan PLTS off Gride Dengan Kapasitas 400 KWP di Pulau Tabuhan PT PLN Persero Unit Induk Distribusi Lampung ". Dan pada tahun 2024 penulis melakukan penelitian dengan judul sekripsi ". Implementasi *Neural Network* Untuk Banyak Rangkaian Kombinasional" dibawah bimbingan Bapak Dr. Eng. Ageng Sadnowo Repelianto, S.T., M.T. dan Ibu Herlinawati, S.T., M.T.

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Alhamduillah, Atas Izin Allah yang Maha Kuasa

Serta dengan mengiringkan shalawat kepada Nabi Muhammad SAW.

Dengan rasa syukur yang mendalam saya persembahkan skripsi ini untuk:

Kedua Orang Tua Saya

Sapto Bugiono

Dan

Nurhidayati

Serta Kepada Adik Tersayang

Alima Miftahul Janah

Yang selalu memberikan doa,dukungan,kasih sayang dan pelajaran hidup yang banyak selama menempuh kuliah di UNILA.

Serta

Keluarga Besar, Dosen Teknik Elektro, Teman Dan Almamater

MOTTO

“Jangan Takut Jatuh, karena yang tidak pernah memanjat yang tidak pernah jatuh. Dan jangan takut gagal, karena yang tidak pernah gagal hanyalah orang-orang yang tidak pernah melangkah. Dan jangan takut salah, karena dengan kesalahan yang pertama kita dapat menambah pengetahuan untuk mencari jalan yang benar pada langkah yang kedua**”**

(Buya Hamka)

“Bukan kesulitan yang membuatnya menakutkan, tetapi ketakutan yang membuatnya sulit."

(Joko widodo)

SANWACANA

Syukur Alhamdulillahirobbilalamin, Penulis haturkan puji syukur atas kehadirat Allah SWT yang senantiasa melimpahkan rahmat dan hidayah, serta inayah-Nya kepada penulis sehingga penulis dapat menyelesaikan laporan sekripsi dengan judul "**Implementasi *Neural Network* Untuk Banyak Rangkaian Kombinasional**". Yang merupakan salah satu syarat untuk memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung.

Pada kesempatan kali ini, pemulis ingin mengucapkan terima kasih kepada :

1. Ibu Prof. Dr. Ir. Lusmeilia Afriani, D.E.A. IPM., ASEAN.Eng., selaku Rektor Universitas Lampung
2. Bapak Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung
3. Bapak Dr. Eng.Ageng Sadnowo R., S.T., M.T., selaku wakil dekan serta pembimbing utama yang telah banyak memberikan waktu, ide pemikiran, semangat, arahan, motivasi, dan pandangan hidup serta banyak hal yang sudah dilakukan kepada penulis pada setiap kesempatan.
4. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung, sekaligus menjadi Dosen pembimbing pendamping yang telah memberikan bimbingan, arahan, motivasi, kepada penulis di setiap kesempatan dengan baik dan ramah.
5. Bapak Sumadi, S.T., M.T., selaku Ketua Program Studi Teknik Elektro Universitas Lampung
6. Bapak Syaiful Alam S.T., M.T., selaku dosen penguji yang telah memberikan arahan, motivasi, agar lebih baik kedepannya.
7. Bapak Dr.ing.Ardian Ulvan S.T., M,SC. selaku dosen pembimbing akademik (PA) yang telah memberikan nasihat, arahan, bimbingan dengan baik dan tulus kepada penulis selama perkuliahan.

8. Bapak dan Ibu Dosen Jurusan Teknik Elektro Fakultas Teknik Universitas Lampung yang telah memberikan pengajaran dan pandangan hidup selama perkuliahan.
9. Seluruh teman-teman Program Studi Sarjana Teknik Elektro UNILA angkatan 2020 untuk kebersamaan yang telah dijalani. Tiada kata yang dapat penulis utarakan untuk mengungkapkan perasaan senang dan bangga menjadi bagian dari angkatan 2020.
10. Kedua orang tua tercinta, ayah dan ibu, yang senantiasa menyayangi dengan penuh kasih, mendidik dengan penuh kesabaran, membimbing dalam setiap langkah kehidupan, berkorban tanpa pamrih demi kebaikan anaknya, serta selalu menyertai setiap doa agar penulis dapat meraih kesuksesan dan kebahagiaan.
11. Keluarga besar yang telah memberikan semangat dalam menjalani perkuliaahan hingga menyelesaikan dunia perkuliahan.
12. Dengan penuh rasa hormat dan penghargaan, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada pemilik NPM 2015041043 atas segala wawasan, masukan, dukungan, pemikiran, dan kontribusi yang diberikan sangat berarti dalam proses penyusunan karya ini.
13. Kepada rekan kepompong yang telah memberikan bantuan dan arahan yang baik untuk menyelesaikan tugas akhir ini.
14. Kepada semua pihak yang telah memberikan motivasi dalam mengerjakan tugas akhir ini

Bandar Lampung , 18 Februari 2025

Akbar Hidayat
NPM.1015031038

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

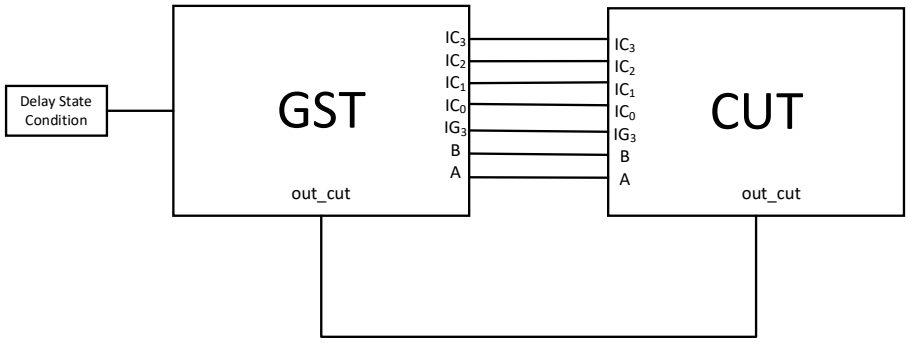
DAFTAR TABEL

Tabel Halaman

BAB I
PENDAHULUAN

1.1. Latar Belakang

Pengujian terhadap produk IC untuk memberikan jaminan pada kualitas produksi. Mekanisme pengujian IC digital umumnya menggunakan rangkaian *Generator Set Test* (GST). Setiap model rangkaian IC digital membutuhkan satu unit *Generator Set Test*. Artinya untuk setiap rangkaian yang berbeda *Generator Set Test* juga berbeda. Secara konsep *Generator Set Test* adalah sebuah fungsi rangkaian digital kombinasional. Gambar 1.1 menunjukkan sistem dari *Generator Set Test* (GST) (Denny Pratama, 2024).



Gambar 1.1. *Generator Set Test* (GST)

Rangkaian digital kombinasional memiliki peran penting dalam berbagai aplikasi elektronik, termasuk pengolahan data, komunikasi, dan sistem kontrol. Namun, keandalan rangkaian ini sangat bergantung pada kemampuan untuk mendeteksi dan mengatasi kesalahan yang mungkin terjadi selama operasinya. Salah satu metode yang umum digunakan untuk mendeteksi kesalahan adalah dengan membangun *Generator Set Test*. *Generator Set Test* (GST) yang dibangun berdasarkan prinsip pohon diagnosa, secara umum bekerja seperti rangkaian digital kombinasional.

Fungsi dari rangkaian digital kombinasional dapat difungsikan dengan *neural network* (Raji, Sabet and Ghavami, 2019).

Penelitian yang sudah dilakukan oleh Denny Pratama, (2024) membangun *Generator Set Test* menggunakan metode *neural network*. Setiap rangkaian elektronika mempunyai pohon uji yang berbeda-beda tergantung dari fungsi rangkaiannya. Ini berarti setiap rangkaian elektronika mempunyai satu set *Generator Set Test* pada setiap rangkaiannya. Hal ini menyebabkan biaya yang besar untuk setiap pengujian pada rangkaian elektronika (Denny Pratama, 2024). Pada rangkaian kombinasional yang berbeda fungsinya, selama masih memiliki jumlah *input* dan *output* yang sama, maka dapat digantikan dengan sebuah struktur *neural network* dengan jumlah *input* dan *output* yang sama juga. Hanya yang membedakannya ada pada bobot jaringan *neural network.*

Penelitian ini bertujuan untuk membangun jaringan syaraf tiruan (*neural network*) untuk digunakan sebagai alternatif menggantikan berbagai jenis fungsi rangkaian kombinasional yang berada di dalam sebuah *Generator Set Test* tanpa harus mengubah struktur *neural network.* Dengan hanya mengubah bobotnya saja struktur *neural network* dapat difungsikan berbagai rangkaian kombinasional dengan catatan *input* dan *output* sama. Fokus utama penelitian adalah untuk menguji apakah hanya dengan mengubah bobot saja, kita dapat menggantikan berbagai fungsi rangkaian kombinasional yang berbeda di dalam *Generator Set Test* dengan *neural network*, tanpa perlu mengubah perangkat keras rangkaian. Pendekatan ini diharapkan dapat mengurangi ketergantungan pada mikrokontroler, terutama ketika perlu berpindah-pindah antar rangkaian kombinasional yang berbeda dalam setiap pengujian, hanya dengan menyesuaikan bobot jaringan syaraf tiruan dalam perangkat lunak.

1.2. Tujuan Penelitian

Adapun tujuan yang ingin dicapai pada penelitian ini adalah sebagai berikut:

1. Mengimplementasikan sebuah struktur *neural network* untuk banyak fungsi rangkaian kombinasional.

2. Membuat algoritma program pergantian bobot *neural network* untuk kebutuhan rangkaian kombinasional yang diuji.

1.3. Rumusan Masalah

Adapun rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana mengimplementasikan *neural network* sehingga dapat menjadi fungsi pada rangkaian kombinasional dengan jumlah *input* dan *output* yang sama?
2. Bagaimana menggunakan sebuah struktur *neural network* untuk menggantikan fungsi beberapa rangkaian kombinasional?

1.4. Batasan Masalah

Adapun batasan masalah yang ditetapkan dalam penelitian ini adalah sebagai berikut:

1. Metode *neural network* yang digunakan adalah *backpropagation*, yang diterapkan pada setiap rangkaian kombinasional dengan 3 *input* dan 1 *output*.

1.5. Manfaat Penelitian

Adapun manfaat yang didapatkan pada penelitian ini adalah sebagai berikut:

1. Penerapan *neural network* untuk rangkaian kombinasional suatu fungsi menjadi lebih mudah dalam implementasinya.
2. Dengan sebuah *neural network* yang terstruktur, dapat diperankan beberapa fungsi rangkaian kombinasional dengan syarat jumlah *input* dan *output* yang sama.

1.6. Hipotesis

Penelitian ini bahwa *neural network* yang dilatih dengan metode *backpropagation* dapat menggantikan fungsi rangkaian kombinasional dengan jumlah *input* dan *output* yang sama. *Neural network* diharapkan memiliki kemampuan untuk mempertahankan kinerja sistem meskipun terdapat variasi atau gangguan pada *input*, dengan kemampuannya dalam mengenali dan menggeneralisasi pola. Selain itu, penerapan *neural network* dalam fungsi rangkaian kombinasional diyakini dapat menyederhanakan desain sistem, mengurangi ketergantungan pada

komponen perangkat keras khusus untuk setiap gerbang logika, dan menghasilkan sistem elektronik yang lebih efisien dan adaptif.

1.7. Sistematika Penulisan

Adapun sistematika penulisan dalam penulisan laporan tugas akhir ini adalah sebagai berikut:

BAB 1 PENDAHULUAN

Pada bab ini berisi tentang latar belakang, tujuan penelitian, perumusan masalah, Batasan masalah, manfaat penelitian, hipotesis, serta sistematika penulisan laporan penelitian.

BAB II TINJAUAN PUSTAKA

Pada bab ini berisi tentang beberapa teori yang mendukung dan refrensi materi yang diperoleh dari berbagai sumber buku, jurnal dan penelitian ilmiah yang digunakan untuk penulisan laporan tugas akhir ini.

BAB III METODOLOGI PENELITIAN

Pada bab ini berisi tentang waktu dan tempat, alat dan bahan, metode penelitian dan pelaksanaan dalam pengerjaan tugas akhir.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini berisi tentang perancangan peralatan dan pembahasan data hasil pengujian alat yang dirancang.

BAB V PENUTUP

Pada bab ini berisi tentang kesimpulan yang didapat dari hasil penelitian yang dilakukan dan saran yang berdasarkan pada hasil data penelitian untuk perbaikan dan pengembangan yang lebih lanjut agar mendapatkan hasil yang lebih baik dari penelitian yang telah dilakukan.

DAFTAR PUSTAKA

Daftar pustaka berisi daftar sumber kutipan teori yang menjadi landasan dan referensi dalam pembahasan.

BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Dalam penelitian yang dilakukan oleh As'ad Shidqy Aziz dkk. (2022) yang berjudul"Model Neuron *Mc Culloch-Pitts* dalam Pengenalan Pola Logika Dasar" bertujuan untuk menyelesaikan logika dasar AND, OR, NAND dan NOR. Hasil dari penerapan jaringan syaraf tiruan model neuron *Mc Culloch-Pitts* dapat digunakan untuk mengenali pola fungsi logika dasar. Penentuan nilai *threshold* dan bobot dalam model ini dilakukan tanpa proses *learning* atau dengan cara analitik (Aziz, 2022).

Dalam penelitian yang dilakukan oleh Rana Sulthanah dkk. (2023) yang berjudul "Penerapan *Metode Mc Culloch-Pitts* menggunakan *Python* untuk pengujian pengenalan pola operator AND, Operator OR, Operator XOR Pada Fungsi Logika" Metode *McCulloch-Pitts* diterapkan untuk pengenalan pola fungsi logika dengan menggunakan bahasa pemrograman *Python*. Tujuan dari penelitian ini adalah untuk menguji kesesuaian nilai hasil uji dengan tabel kebenaran pada fungsi logika, yang melibatkan operator AND, OR, dan XOR. Pengujian ini melibatkan variabel *input* (Sulthanah and Ahmad, 2023).

Dalam penelitian yang dilakukan oleh Denny Pratama, (2024) yang berjudul "Generator Set Uji Untuk Diagnosis Kesalahan Rangkaian Kombinasional Multiplexer 4 *Line To* 1 *Line* Menggunakan Metode *Artificial Neural Network*". Tujuan dari penelitian ini adalah memanfaatkan *neural network* untuk melakukan proses diagnosis kesalahan dengan menjalankan tahapan pengujian pohon diagnosa pada rangkaian digital kombinasional.

2.2. Rangkaian Gerbang Logika

Gerbang logika merupakan suatu entitas dalam elektronika dan matematika *boolean* yang mengkonversi satu atau beberapa *input* logika menjadi sinyal *output logic*. Gerbang logika diimplementasikan secara elektronik menggunakan dioda atau transistor, akan tetapi dapat pula dibangun menggunakan susunan komponen-komponen yang memanfaatkan sifat-sifat elektromaknetik. Gerbang logika juga dikenal sebagai *logic gate* dalam bahasa Inggris, merupakan elemen dasar dalam pembentukan sistem elektronika digital yang berfungsi mengubah satu atau beberapa *input* menjadi sinyal *output* yang bersifat logis. Operasional gerbang logika didasarkan pada sistem bilangan biner, yang menggunakan hanya dua simbol yaitu 0 dan 1, dengan menerapkan teori aljabar *boolean.* Implementasi gerbang logika dalam sistem elektronika digital melibatkan komponen-komponen elektronika seperti *integrated circuit* (Parinduri and Nurhabibah Hutagalung, 2019).

Logika adalah suatu dasar untuk menyatukan beberapa gerbang logika dimana membutuhkan operator logika dan untuk membuktikan kebenaran dari gerbang logika menggunakan tabel kebenaran. Tabel kebenaran menunjukkan hubungan antara nilai kebenaran dari proposisi atomik. Melalui tabel kebenaran, nilai kebenaran suatu persamaan logika atau proposisi dapat ditemukan. Tabel kebenaran memiliki berbagai aplikasi yang dapat diterapkan karena fungsi dasarnya. Salah satu aplikasi utamanya adalah dalam merancang rangkaian logika. Gerbang logika merupakan komponen dasar elektronik yang berfungsi untuk menghasilkan sinyal *output* berdasarkan sinyal *input* yang diterimanya (Parinduri and Nurhabibah Hutagalung, 2019).

2.3. Macam Macam Rangkaian Dasar Gerbang Logika

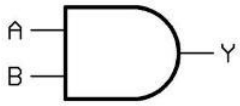
Terdapat berbagai jenis gerbang logika, termasuk gerbang AND, gerbang NAND, gerbang OR, gerbang NOT, gerbang XOR, gerbang NOR, dan gerbang XNOR.

2.3.1. Gerbang AND

Gerbang AND adalah suatu gerbang logika yang memiliki 2 buah *input* atau lebih dan sebuah saluran *output*. Gambar 2.1 menyatakan gerbang AND akan mengeluarakan *output* biner tergantung sesuai dengan kondisi *input* dan fungsinya.

Prinsip kerja dari gerbang logika AND yaitu apabila *output* berlogika 1 bila semua *output* berlogika 1. Selain *input* selain itu maka *output* akan berlogika 0 (Parinduri and Nurhabibah Hutagalung, 2019).

Persamaan aljabar untuk sebuah gerbang logika AND dituliskan pada Persamaan (2.1).



Gambar 2.1. Simbol Gerbang Logika AND

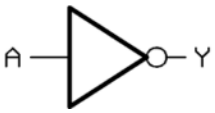
$$Y = A.B \tag{2.1}$$

Tabel 2.1. Tabel Kebenaran Gerbang Logika AND

Input		*Output*
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

2.3.2. Gerbang Logika NOT

Gerbang logika NOT atau sering disebut juga dengan pembalik (*inverter*) yang memiliki fungsi membalik logika tegangan *input* pada *output*. Pada logika tegangan hanya ada dua kondisi yaitu dimana pada kondisi 1 (*high*) dan kondisi 0 *(low*). Maka dalam membalik logika mengubah 1 menjadi 0 atau sebaliknya mengubah 0 menjadi 1. Gambar 2.2 menunjukkan simbol atau tanda pada gerbang logika.



Gambar 2.2. Simbol Gerbang Logika NOT

Persamaan aljabar untuk sebuah gerbang logika NOT dituliskan pada Persamaan (2.2).

$$Y = \overline{A} \tag{2.2}$$

Tabel 2.2. Tabel Kebenaran Gerbang Logika NOT

Input	*Output*
A	Y
0	1
1	0

2.3.3. Gerbang Logika NAND

Gerbang NAND merupakan bentuk pengembangan dari gerbang AND. Pada dasarnya, gerbang ini adalah gerbang AND yang *input*-nya dihubungkan dengan gerbang NOT. Gambar 2.3 yang menunjukkan kombinasi kedua gerbang tersebut dan simbol dari gerbang NAND. Tabel kebenaran gerbang NAND, yang merupakan kebalikan dari tabel kebenaran gerbang AND.

Gambar 2.3. Simbol Gerbang Logika NAND

Persaman aljabar *boolean* untuk sebuah gerbang logika OR dituliskan pada Persamaan (2.3).

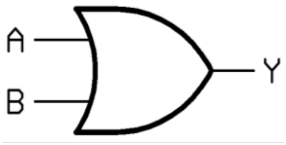
$$Y=\overline{A}\,.\,\overline{B} \qquad (2.3)$$

Tabel 2.3 Tabel Kebenaran Gerbang Logika NAND

Input		*Output*
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

2.3.4. Gerbang Logika OR

Gerbang logika OR merupakan gerbang logika dasar yang memiliki 2 buah saluran *input* atau lebih dan 1 buah saluran *output*. Berapapun jumlah saluran *input* yang dimiliki oleh sebuah gerbang OR yang ditunjukkan pada Gambar 2.4 prinsip kerjanya tetap sama. *Output* akan bernilai logika 1 jika salah satu atau semua saluran *input* bernilai logika 1. Sebaliknya, *output* akan bernilai logika 0 ditunjukkan pada Tabel 2.4 (Sugiartowo and Ambo, 2018).



Gambar 2.4. Simbol Gerbang Logika OR

Persaman aljabar *boolean* untuk sebuah gerbang logika OR dituliskan pada Persamaan (2.4).

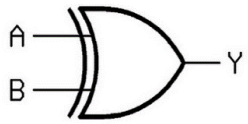
$$Y = A + B \quad (2.4)$$

Tabel 2.4. Tabel Kebenaran Gerbang Logika OR

Input		*Output*
A	B	Y
0	0	0
1	0	1
0	1	1
1	1	1

2.3.5. Gerbang Logika XOR

Gerbang logika XOR atau dikenal dengan gerbang *exclusive* OR dimana jika *input* berlogika sama maka *output* akan berlogika 0 dan sebaliknya jika *input* berlogika berbeda maka *output* akan berlogika 1. Pada Tabel 2.5 tabel kebenaran gerbang logika XOR (Jan, Sianipar and Sultan, 2015). Gambar 2.5 menunjukkan simbol atau tanda pada gerbang logika XOR (Sugiartowo and Ambo, 2018).



Gambar 2.5. Simbol Gerbang Logika XOR

Persaman aljabar *boolean* untuk sebuah gerbang logika XOR dituliskan pada Persamaan (2.5).

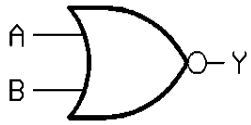
$$Y = A \oplus B \quad (2.5)$$

Tabel 2.5. Tabel Kebenaran Gerbang Logika XOR

Input		*Output*
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

2.3.6. Gerbang Logika NOR

Gerbang logika NOR adalah gerbang logika perkembangan dari gerbang OR yang berupa pemasangan gerbang NOT pada *input* dari gerbang OR. Pada Gambar 2.6 menunjukkan gabungan gabungan OR dan NOT beserta simbol dari gerbang NOR. Pada dasarnya gerbang OR yang *input* dibalik maka pada tabel *output* adalah kebalikan dari gerbang OR ditunjukkan pada Tabel 2.6 (Sugiartowo and Ambo, 2018).



Gambar 2.6. Gerbang Logika NOR

Persaman aljabar *Boolean* untuk sebuah gerbang logika NOR dituliskan pada Persamaan (2.6).

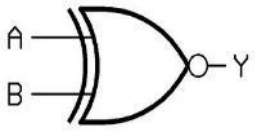
$$Y = \overline{A} + \overline{B} \quad (2.6)$$

Tabel 2.6. Tabel Kebenaran Gerbang Logika NOR

Input		*Output*
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

2.3.7. Gerbang Logika XNOR

Gerbang Logika XNOR adalah singkatan dari *exclusive* NOT OR adalah kebalikan dari gerbang XOR dimana *input* berlogika sama maka *output* atau *input* akan berlogika 1 dan sebaliknya jika *input* atau *output* berlogika beda maka hasil *output* akan berlogika 0 seperti yang ditunjukkan pada Tabel 2.7. Untuk simbol gerbang XNOR tesebut ditunjukkan pada Gambar 2.7.



Gambar 2.7. Gerbang Logika XNOR

Persaman aljabar *Boolean* untuk sebuah gerbang logika XNOR dituliskan pada Persamaan (2.7).

$$Y = \overline{A} \oplus \overline{B} \quad (2.7)$$

Tabel 2.7. Tabel Kebenaran Gerbang Logika XNOR

Input		*Output*
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

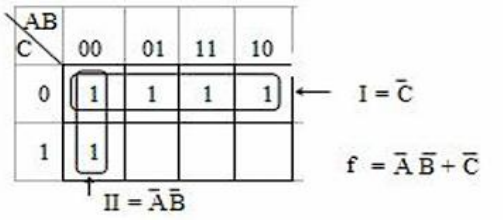
2.4. Rangkaian Kombinasional

Rangkaian kombinasional merupakan suatu rangkaian yang terdiri dari beberapa gerbang logika dasar dengan nilai *output* tergantung nilai *input*. Rangkaian kombinasional tidak memiliki sifat penyimpanan, maka nilai *output* rangkaian pada waktu tersebut hanya ditentukan oleh nilai dari *input* di waktu tersebut. Rangkaian logika kombinasional terdiri dari berbagai gerbang logika, sehingga sangat kompleks dan rumit. Namun, ada beberapa cara untuk menyederhanakannya.

Dalam penyederhanaan rangkaian logika diantaranya menggunakan metode *Karnaugh Map*, *De Morgan* dan metode Aljabar *Boolean* dijelaskan sebagai berikut:

1.Metode *Karnaugh Map*

Metode ini digunakan untuk menyederhanakan fungsi *Boolean*. Dengan cara memisakan tabel kebenaran dengan kotak-kotak segi empat yang jumlahnya tergantung dari jumlah peubah (variabel) masukan penyederhanaan setiap satu (1) yang bertentangan 2,4,8,16,… menjadi suku mintrm yang sederhana (Indrawaty, Kristina and Nugraha, 2012).



Gambar 2.8. Metode Karnaugh Map

2.Metode *De Morgan*

Hukum *De Morgan* menyatakan bahwa operator NOT diterapkan dari 2 variabel AND atau sama dengan diterapkan pada masing masing dari 2 variabel NOT

dengan OR diantaranya. Contoh 2 pada Persamaan (2.8) ini dikenal dengan nama hukum *De morgan*:

$$(A.B.C) = A + B + C \quad (2.8)$$

$$(\mathrm{A} + \mathrm{B} + \mathrm{C}) = A.B.C \quad (2.9)$$

3.Metode Aljabar *Boolean*

Aljabar *Boolean* memuat variable dan simbol operasi untuk gerbang logika. Simbol yang diterapkan pada Ajabar *Boolean* adalah (.) untuk AND, (+) untuk OR, dan ($\bar{A}$) untuk NOT. Rangkaian logika merupakan kombinasi dari berbagai gerbang. Sifat-sifat Aljabar *Boolean* digunakan untuk mempermudah penyelesaian perhitungan aljabar dengan mengisi tabel kebenaran. (Indrawaty, Kristina and Nugraha, 2012). Dalam Aljabar *Boolean*, terdapat dua konstanta utama, yaitu logika 0 dan logika 1. Ketika kedua logika ini diimplementasikan dalam rangkaian logika, mereka akan merepresentasikan tingkat tegangan tertentu. Logika 0 biasanya diwakili oleh tegangan rendah, sedangkan logika 1 diwakili oleh tegangan tinggi. Teori-teori dalam Aljabar *Boolean* ini didasarkan pada aturan-aturan dasar yang mengatur hubungan antara variabel-variabel *Boolean*. Dalil-dalil *Boolean* (*Boolean postulates*) sebagai berikut:

P1: X= 0 atau X=1 P2: 0 . 0 = 0

P3: 1 + 1 = 1 P4: 0 + 0 = 0

P5: 1 . 1 = 1 P6: 1 . 0 = 0 . 1 = 0

P7: 1 + 0 = 0 + 1 = 1

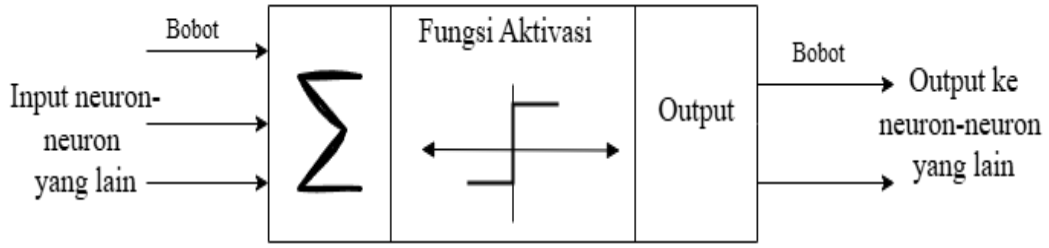
2.5. Jaringan Syaraf Tiruan (JST)

2.5.1. Pengertian Jaringan Syaraf Tiruan (JST)

Jaringan syaraf tiruan adalah metode pengolahan informasi yang terinspirasi oleh sistem syaraf biologis, mirip dengan cara kerja otak manusia. Jaringan syaraf tiruan yaitu upaya manusia untuk meniru fungsi dari sistem syaraf manusia dalam memproses informasi untuk melaksanakan berbagai tugas. Jaringan syaraf tiruan ini terdiri dari sejumlah neuron yang terhubung oleh jaringan. Setiap jaringan memiliki bobot yang digunakan untuk mengirimkan informasi dari satu neuron ke neuron lain yang terhubung. Bobot ini akan terus diperbarui selama proses pembelajaran (Suryadibrata and Chandra, 2020).

2.5.2. Komponen Jaringan Syaraf Tiruan

Terdapat berbagai macam tipe jaringan syaraf, dan setiap jenis jaringan umumnya memiliki komponen yang serupa. Jaringan syaraf juga tersusun dari beberapa neuron, mirip dengan cara kerja otak manusia. Neuron-neuron ini akan mengubah data yang diterima melalui koneksi *output* dan mengirimkannya ke neuron lainnya. Dalam jaringan syaraf, hubungan ini disebut sebagai bobot. Data tersebut disimpan dalam suatu nilai spesifik pada bobot tersebut. Pada Gambar 2.9 merupakan struktur neuron jaringan syaraf tiruan (Wuryandari and Afrianto, 2012).



Gambar 2.9. Struktur Neuron Jaringan Syaraf Tiruan

Sebuah sel syaraf pada jaringan syaraf buatan terdiri dari tiga komponen, yaitu penjumlahan fungsi, fungsi aktivasi, dan hasil. Fungsi penjumlahan berperan dalam mengakumulasi nilai bobot pada setiap jaringan yang terhubung dengan neuron. Sedangkan fungsi aktivasi berfungsi untuk mengaktifkan neuron atau menentukan output dari neuron tersebut. Informasi *input* pada jaringan syaraf tiruan dikirimkan ke setiap neuron dengan nilai bobot tertentu. Nilai bobot tersebut dijumlahkan oleh fungsi penjumlahan dan hasil penjumlahan tersebut dibandingkan dengan nilai ambang (*threshold)* tertentu melalui fungsi aktivasi setiap neuron. Jika *input* melebihi nilai ambang tersebut, neuron akan diaktifkan, jika tidak neuron tidak akan diaktifkan. Jika neuron diaktifkan, maka akan mengirimkan *output* melalui bobot-bobot *output*nya ke semua neuron yang terhubung dengannya, dan proses ini berlanjut seterusnya (Wahyono, 2011). Pembagian arsitektur jaringan syaraf tiruan dapat dilihat dari *framework* dan skema koneksi. Kerangka jaringan syaraf tiruan dapat dikenali dari jumlah lapisan dan *node* pada setiap lapisan.

Lapisan yang membentuk jaringan syaraf tiruan dapat dibagi menjadi tiga lapisan yaitu lapisan *input (Input Layer)*, lapisan tersembunyi (*Hidden Layer)*, dan lapisan *output (Output Layer)* adalah sebagai berikut:

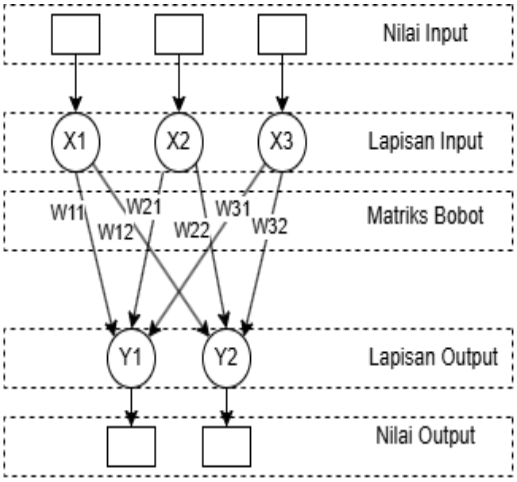
1. *Input layer* disebut sebagai lapisan *input node-node* di dalam lapisan *input*. Unit unit *input* menerima *input* dari luar. *Input* yang dimasukkan adalah suatu penggambaran masalah.
2. *Hidden layer* adalah suatu *node-node* di dalam *hidden layer*. *Output* dari *hidden layer* tidak bisa langsung terlihat secara langsung.
3. *Output layer* adalah suatu *node-node* di dalam *output layer*. *Output* dari lapisan ini adalah suatu *output* Jaringan Syaraf Tiruan (JST) terhadap suatu permasalahan (Hadimarta, Muhima and Kurniawan, 2020).

2.5.3. Arsitektur Jaringan Syaraf Tiruan

Seperti yang dijelaskan sebelumnya bahwa neuron-neuron dikelompokan dalam lapisan-lapisan. Umumnya neuron-neuron yang terletak pada lapisan yang sama akan memiliki keadaan yang sama. Faktor terpenting untuk menentukan kelakuan suatu neuron adalah fungsi aktivasi dan pola bobot (Wuryandari and Afrianto, 2012). Pada setiap lapisan yang sama, neuron-neuron memiliki fungsi aktivasi yang sama. Ada beberapa arsitektur jaringan syaraf antara lain:

2.5.3.1.Jaringan Lapisan Tunggal (*Single Layer*)

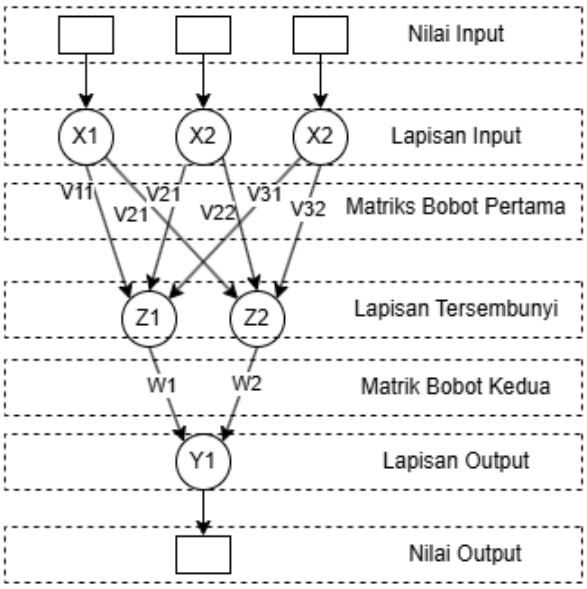
Dalam struktur jaringan satu lapisan (*Single Layer*), terdapat hanya satu lapisan dengan bobot yang berhubungan. Jaringan ini hanya menerima *input* dan langsung mengolahnya menjadi *output* tanpa melewati lapisan tersembunyi (*hidden layer*) (Wuryandari and Afrianto, 2012). Pada Gambar 2.10 menunjukkan bahwa lapisan *input* memiliki 3 neuron yaitu X1, X2, X3. Sedangkan pada lapisan *output* memiliki 2 neuron yaitu Y1 Dan Y2. Neuron-neuron pada lapisan tersebut saling berhubungan seperti yang ditunjukkan pada Gambar 2.10.



Gambar 2.10. Jaringan *Single Layer*

2.5.3.2. Jaringan Dengan Banyak Lapisan (*Multilayer*)

Jaringan banyak lapisan (*Multilayer*) memiliki satu atau banyak lapisan yang terletak diantara lapisan *input* dan lapisan *output* seperti yang ditunjukkan pada Gambar 2.11. Biasanya, terdapat lapisan bobot yang terletak diantara dua lapisan yang bersebelahan. Jaringan *multilayer* mampu mengatasi masalah yang lebih kompleks dan memerlukan pembelajaran yang lebih rumit dibandingkan dengan jaringan *single layer*. Meskipun membutuhkan waktu lebih lama, jaringan ini lebih efektif dalam menyelesaikan permasalahan (Paramita, 2010).



Gambar 2.11. Jaringan *Multilayer*

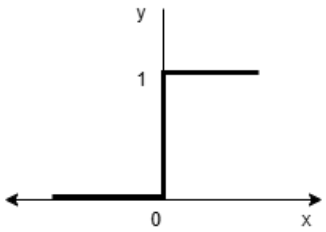
2.5.4. Fungsi Aktivasi

Berikut ini merupakan beberapa fungsi aktivasi yang sering digunakan dalam jaringan syaraf tiruan adalah sebagai berikut:

a) Fungsi Undak Biner (*Hard Limit*)

Pada jaringan lapis tunggal digunakan fungsi *step function* atau undak untuk mengkonversi *input* dari satu variabel yang bernilai kontinu ke suatu *output* biner yang dirumuskan dengan Persamaan (2.10).

$$y = \begin{cases} 0, jika\ x \leq 0 \\ 1, jika\ x > 0 \end{cases} \quad (2.10)$$

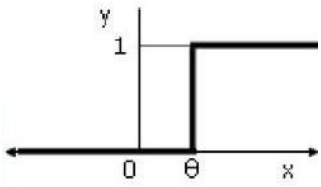


Gambar 2.12. Fungsi Aktvasi Undak Biner

b) Fungsi Undak Biner (*Threshold)*

Fungsi undak biner dengan menggunakan nilai ambang atau sering disebut dengan nilai ambang (*threshold)*. Fungsi undak biner dirumuskan pada Persamaan (2.11).

$$y = \begin{cases} 0, jika\ x\ < \theta \\ 1, jika\ x \geq \theta \end{cases} \quad (2.11)$$

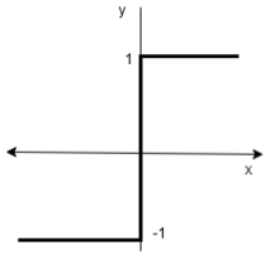


Gambar 2.13. Fungsi Aktivasi Undak Biner (*Threshold*)

c) Fungsi Aktivasi Bipolar (*Symetric Hard Limit*)

Fungsi aktivasi bipolar hampir sama dengan fungsi aktivasi undak biner, hanya saja yang membedakan *output* yang dihasilkan berupa 0, 1 atau -1. Fungsi *Symetric Hard Limit* dituliskan pada Persamaan (2.12).

$$y = \begin{cases} 1, jika\ x > 0 \\ 0, jika\ x = 0 \\ -1, jika\ x < 0 \end{cases} \quad (2.12)$$

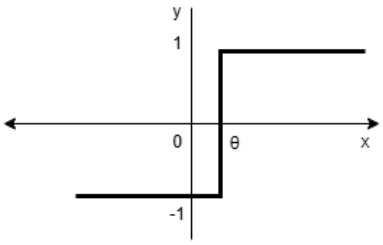


Gambar 2.14. Fungsi Aktivasi Bipolar (*Symetric Hard Limit*)

d) Fungsi Bipolar (Dengan *Threshold*)

Fungsi bipolar hampir sama dengan fungsi aktivasi undak biner *threshold*, hanya saja *output* yang dihasilkan berupa 1, 0 atau -1 yang ditunjukkan pada Gambar 2.15 serta dituliskan rumusnya pada Persamaan (2.13).

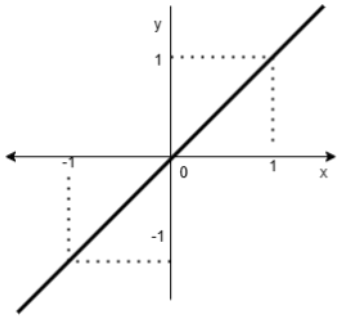
$$y = \begin{cases} -1, jika\ x\ \geq \theta \\ 1, jika\ x < \theta \end{cases} \quad (2.13)$$



Gambar 2.15. Fungsi Aktivasi Bipolar Dengan *Threshold*

e) Fungsi *Linear* (Identitas)

Fungsi *linear* identitas memiliki nilai *output* yang sama dengan nilai *input* yang ditunjukkan pada grafik Gambar 2.16.

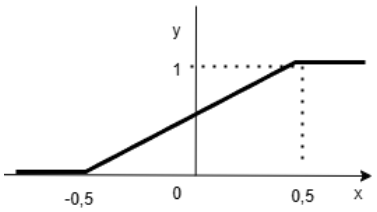


Gambar 2.16. Fungsi *Linear* Identitas

f) Fungsi *Saturating Linear*

Fungsi *saturating linier* ini hanya akan bernilai 0 jika *input* kurang dari -0,5 dan akan bernilai satu jika *input* lebih dari 0,5. Sedangkan jika nilai *input* terletak pada -0,5 dan 0,5 maka *output* yang akan dihasilkan sama dengan nilai *input* ditambah 0,5 seperti grafik Gambar 2.17 dengan dirumuskan pada Persamaan (2.14).

$$y = \begin{cases} 1\ , jika\ x \geq 0{,}5 \\ x + 0{,}5\ , jika\ -0{,}5 \leq x \leq 0{,}5 \\ jika\ x < -0{,}5 \end{cases} \quad (2.14)$$

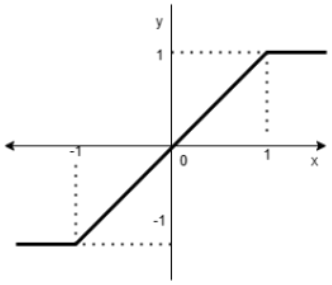


Gambar 2.17. Fungsi *Saturating Linear*

g) Fungsi *Symetric Saturating Linear*

Pada fungsi ini akan bernilai -1 jika *input* kurang dari -1, dan akan bernilai 1 jika *input* lebih dari 1. Sedangkan jika *input* terletak antara -1 dan 1 maka *output* akan bernilai sama dengan dengan nilai *input* seperti yang ditunjukkan pada Gambar 2.18 dengan dirumuskan Persamaan (2.15) (Paramita, 2010).

$$y = \begin{cases} 1, jika\ x \geq 1 \\ x, jika\ x = -1 \leq x \leq 1 \\ -1, jika\ x \leq -1 \end{cases} \quad (2.15)$$



Gambar 2.18. Fungsi Aktifasi Fungsi *Symetric Saturating Linear*

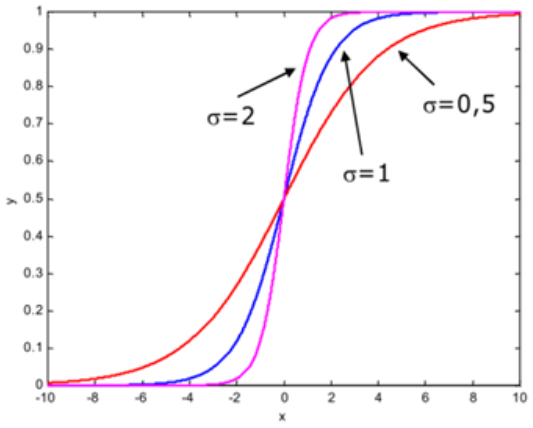
h) Fungsi *Sigmoid* Biner

Fungsi *sigmoid* biner digunakan untuk jaringan syaraf tiruan yang dilatih menggunakan metode *backpropagation*. Fungsi *sigmoid* biner memiliki nilai pada range 0 sampai 1. Fungsi ini sering digunakan untuk jaringan syaraf yang membutuhkan nilai *output* yang terletak pada interval 0 sampai 1. Namun fungsi ini bisa juga digunakan oleh jaringan syaraf yang memiliki *output* 0 atau 1 (Pamungkas, Sumadi and Alam, 2022). Fungsi *sigmoid* biner ini ditunjukkan pada Gambar 2.19 yang dirumuskan pada Persamaan (2.16).

$$y = f(x) = \frac{1}{1+e^{-\sigma x}} \tag{2.16}$$

dengan turunan :

$$f'(x) = \sigma f(x)[1 - f(x)] \tag{2.17}$$



Gambar 2.19. Fungsi Aktivasi *Sigmoid* Biner

2.5.5. *Backpropagation*

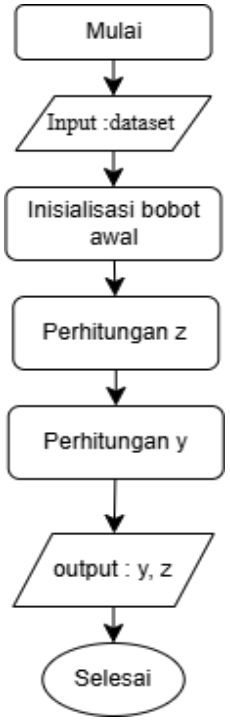
2.5.5.1. Pengertian Metode *Backpropagation*

Backpropagation adalah salah algoritma pembelajaran yang terawasi dan biasanya digunakan oleh perceptron dengan banyak lapis. Jaringan ini sering digunakan dalam implementasi dengan banyak lapis untuk mengubah bobot yang terhubung dengan neuron yang ada pada *hidden layer* (Salsabila and Cholissodin, 2020). Secara garis besar dengan berdasarkan proses *training* metode *backpropagation* dibagi menjadi 2 bagian yaitu metode *feedforward* (tahap maju) dan *backpropagation* (propagasi balik) sebagai berikut:

a) Metode *feedforward* (propagasi maju)

Feedforward atau disebut dengan tahap maju merupakan suatu proses pengolahan *input* pada lapisan *input* hingga respon yang didapatkan disalurkan ke lapisan *output*. Fase ini meliputi proses inisialisasi bobot awal, proses menghitung jumlah dari *input* ($zinj$) dan *hidden layer* ($yink$) serta menghitung nilai aktivasi dari *input* (zj) dan dari *hidden layer* (yk). Gambar 2.20 merupakan diagram alir fase *feedforward* yang dijelaskan sebagai berikut:

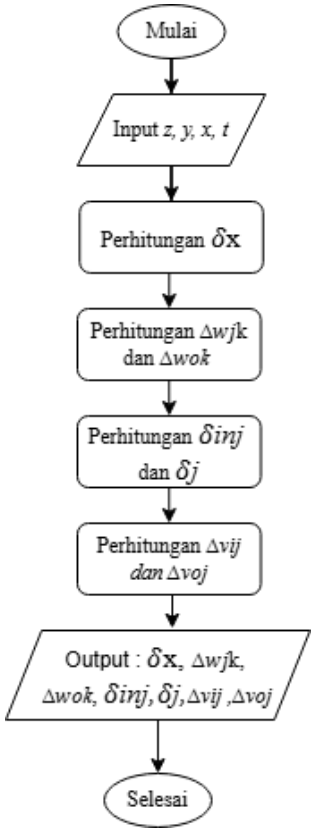
a) Memasukkan *input* kepada sistem berupa *dataset* yang digunakan.
b) Proses inisialisasi bobot awal dengan nilai acak (v, w).
c) Menghitung nilai z, yang terdiri dari nilai yang diterima dari *input* ($zinj$) dan nilai aktivasinya (zj).
d) Menghitung nilai y, terdiri dari nilai yang diterima dari *hidden layer*
e) ($yink$), dan nilai aktivasinya (yk).
f) Proses *output* yang menghasilkan nilai z dan y.



Gambar 2.20. Proses *Feedforward*

b) *Backpropagation* (propagasi balik)

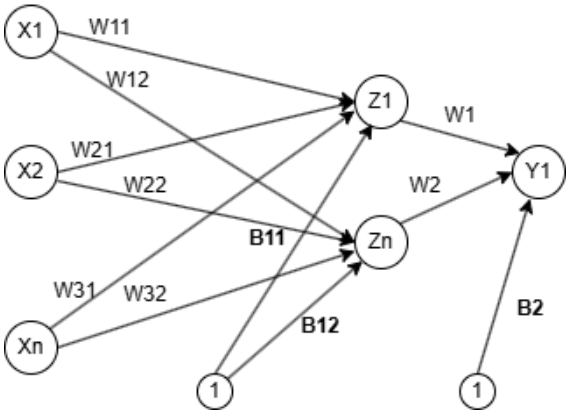
Fase *backpropagation* atau propagasi balik adalah hasil pada lapisan *output* yang akan dibandingkan dengan target *output*, lalu di hitung nilai *error* (δ). Jika iterasi atau kondisi berhenti belum terpenuhi maka akan dilanjutkan ke tahap berikutnya. Jika sudah terpenuhi maka proses perhitungan akan berhenti. Pada Gambar 2.21 merupakan diagram alir dengan fase *backpropagation* menurut fussett (1994).



Gambar 2.21. Proses *Backpropagation*

2.5.5.2. Jaringan Metode *Backpropagation*

Backpropagation merupakan salah satu metode pelatihan dari jaringan syaraf tiruan. Pada Gambar 2.22 menyatakan *backpropagation* menggunakan arsitektur *multilayer* dengan metode *surpervised training*.



Gambar 2.22. Arsitektur *Multilayer*

Keterangan:

X= *Input*
W=Bobot pada lapisan
B= *Bias* pada lapisan
Y=*Output* hasil

Tahap ini adalah pengenalan pola data yang telah dinormalisasi agar sistem dapat menentukan yang dapat memetakan antara data *input* dengan data target *output* yang diinginkan (Rachman, Cholissodin and Fauzi, 2018) .
Tahap-tahap pelatihan untuk jaringan *backpropagation* sebagai berikut:

Fase 1: *feedforward*

1. Melakukan inisialisasi nilai bobot dan *bias* diatur dengan nilai acak antar -0,5 hingga 0,5. Inisialisasi nilai *learning rate*, nilai maksimal iterasi dan nilai minimum *error* ditentukan oleh pengguna sistem.
2. Melakukan peroses pelatihan selama syarat berhenti masih belum terpenuhi. Nilai maksimal iterasi atau minimum *error* adalah penentu syarat berhenti. Pelatihan berhenti jika sudah melebihi iterasi maksimal.
3. Masing-masing unit *input* x_i menerima sinyal *input* dan dikirim ke seluruh *hidden layer*.
4. Masing-masing *hidden* akan diproses melalui sinyal *input* dengan bobot dan *bias* berdasarkan Persamaan (2.18).

   $$Z_{in_j} = vo_j + \sum_{i=1}^{n} x_i \, v_{ij} \quad (2.18)$$

5. Lalu fungsi aktivasi yang telah ditentukan memperoleh sinyal *output* berdasarkan *hidden* unit seperti yang ditunjukkan pada Persamaan (2.19)

   $$z_j = f(z_{inj}) = \frac{1}{(1+(exp^{-zinj}))} \quad (2.19)$$

   Nilai *output* y_k akan menghitung seperti Persamaan (2.20).

   $$y - in_k = vo_k + \sum_{j=1}^{p} z_j \, w_{jk} \quad (2.20)$$

   Kemudian fungsi aktivasi menghitung nilai *output* berdasarkan Persamaan (2.21).

$$y_k = f(y_{in_k}) = \frac{1}{(1+(exp^{-y_{ink}}))} \quad (2.21)$$

Fase 2: *Backpropagation*

6. Melakukakan perhitungan koreksi *error* (δ_k) menggunakan Persamaan (2.22).

   $$\delta_k = (t_k - y_k)y_k(1 - y_k) \quad (2.22)$$

   Hasil dari faktor koreksi *error* diteruskan ke proses perhitungan koreksi bobot. ΔW_{jk} untuk memperbarui ΔW_{jk}. Proses pembaruan nilai ΔW_{jk} ditunjukkan seperti Persamaaan (2.23).

   $$\Delta W_{jk} = \propto \delta_k z_j \quad (2.23)$$

   Jika iterasi pertama, Persamaan yang dipakai seperti Persamaan (2.24).

   $$\delta\_in_{j=} \sum_{k=1}^{m} \delta_k \, w_{jk} \quad (2.24)$$

   Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi untuk mendapatkan faktor koreksi *error* pada Persamaan (2.25) dan (2.26).

   $$\delta_j = \delta_{inj} z_j (1 - z_j) \quad (2.25)$$

   $$\Delta V_{ij} = \delta_j x_i \quad (2.26)$$

   Pada iterasi yang kedua dan selanjutnya pada Persamaan (2.27)

   $$\delta\_in_{j=} \sum_{k=1}^{m} \delta_k \, w_{jk} \quad (2.27)$$

   Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi untuk mendapatkan faktor koreksi *error* pada Persamaan (2.28) dan (2.29).

   $$\delta_j = \delta_{inj} z_j (1 - z_j) \quad (2.28)$$

   $$\Delta V_{ij} = (m\delta_{j-1}) + (\propto \delta_j x_i) * (1 - m) \quad (2.29)$$

Fase 3: Perubahan Bobot

7. Setiap unit *output* (Y_k,k=1,….m) akan mempengaruhi bobotnya dari setiap *hidden* unit pada Persamaan (2.30).

   $$\Delta W_{jk}(baru) = \Delta W_{jk}(lama) + \Delta W_{jk} \quad (2.30)$$

   Demikian pula setiap *hidden* unit (Z_j,j=1,….p) dari setiap unit *input* akan mempengaruhi bobotnya pada Persamaan (2.31).

   $$V_{ij}(baru) = V_{ij}(lama) + \Delta V_{ij} \quad (2.31)$$

8. Memeriksa *stop condition,* jika nilai *error* sudah kecil dari nilai minimum *error* atau iterasi sudah mencapai maksimum maka proses pelatihan dihentikan

(Pamungkas, Sumadi and Alam, 2022). Perhitungan nilai *error* pada pelatihan menggunakan *Mean Square Error* (MSE) seperti Persamaan (2.32).

$$MSE = \frac{\sum_{i=1}^{n}(T_i - T'_i)}{n} \quad (2.32)$$

Keterangan

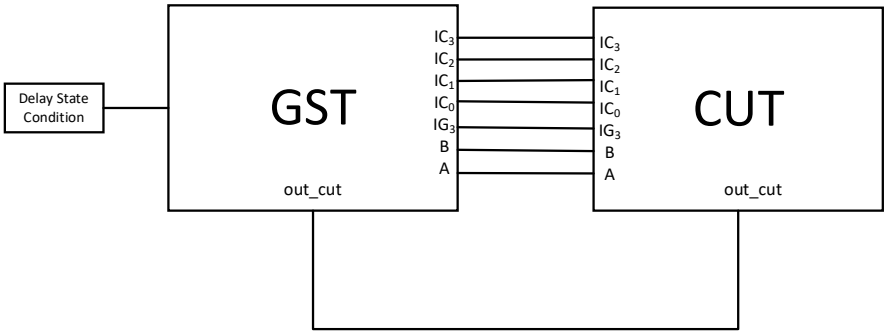
T_i = Nilai actual pada data ke-i

T'_i = Nilai hasil peramalan untuk data ke-i

n = Banyaknya data

2.6. *Generator Set Test* (GST)

Generator Set Test (GST) adalah sistem yang menghasilkan pola sinyal uji untuk menguji rangkaian digital, seperti rangkaian kombinasi atau sekuensial. GST mengirimkan sinyal uji ke *Circuit Under Test* (CUT) dan membandingkan *output* dengan hasil yang diharapkan untuk mendeteksi kesalahan. Dilengkapi mekanisme umpan balik, GST dapat menyesuaikan pola uji secara adaptif, meningkatkan efisiensi dan akurasi pengujian. GST sering digunakan dalam pengujian rangkaian logika, validasi desain, dan diagnosis kerusakan elektronik.Rangkaian GST ditunjukkan pada Gambar 2.23 (Denny Pratama, 2024).



Gambar 2.23. *Generator Set Test*

2.7. *Python*

Python adalah bahasa pemrograman serbaguna yang menggunakan pendekatan interpretatif dan menekankan pada keterbacaan kode. *Python* dikenal sebagai bahasa yang menggabungkan kapabilitas dan fleksibilitas dengan sintaksis yang sangat jelas (Hasibuan *et al.*, 2024). Bahasa pemrograman *Python*, yang saat ini sangat populer, pertama kali diciptakan oleh *Guido van Rossum di Stichting*

Mathematisch Centrum (CWI) di Amsterdam pada tahun 1991. Salah satu perbedaan utama *Python* dengan bahasa pemrograman lainnya adalah bahwa pengembangannya melibatkan jutaan programmer, peneliti, dan pengguna dari berbagai latar belakang, tidak hanya dari kalangan IT, karena *Python* bersifat *open source*. Berikut merupakan logo dari *Python* ditampilkan pada Gambar 2.24 (Rahman *et al.*, 2023).

Gambar 2.24. Logo *Python*

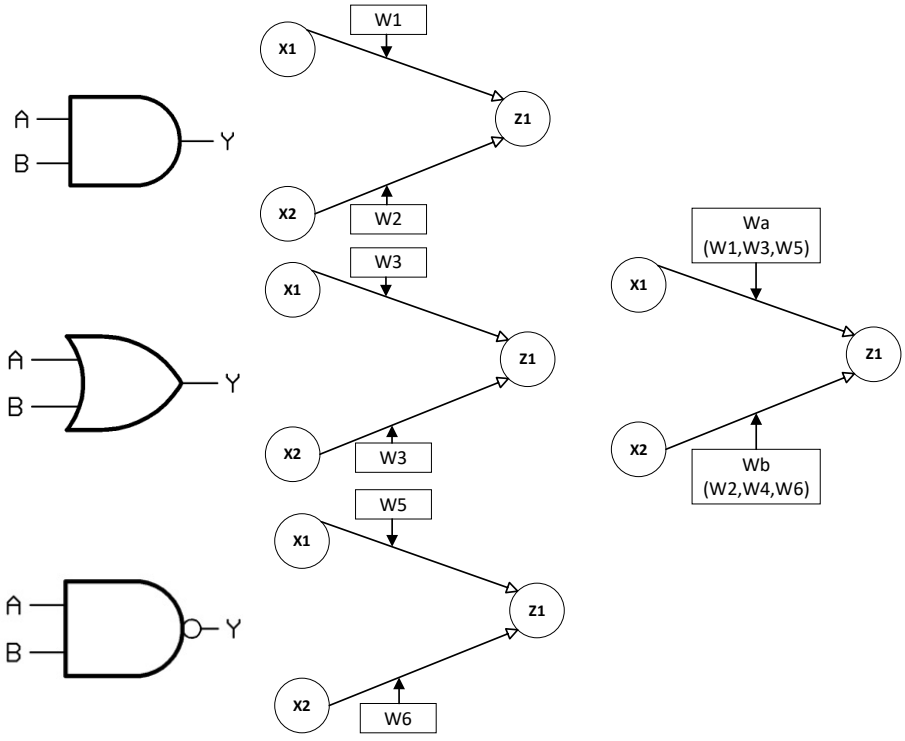
BAB III
METODOLOGI PENELITIAN

3.1. Waktu Dan Tempat

Adapun penelitian ini yang dilakukan di Laboratorium Teknik Elektronika, Laboratorium Terpadu Jurusan Teknik Elektro, Universitas Lampung, dimulai pada bulan Mei 2024 sampai bulan Februari 2025.

3.2. Arsitektur Konsep Sistem Perancangan

Adapun arsitektur konsep sederhana sistem perencanaan dasar gerbang logika AND, OR, dan NAND yang diimplementasikan ke dalam arsitektur *neural network* seperti yang ditunjukkan pada gambar 3.1.



Gambar 3.1. Arsitektur Konsep Sistem Perancangan

Berdasarkan konsep sistem perancangan sederhana yang ditunjukkan pada Gambar 3.1, menunjukkan tiga gerbang logika AND, OR dan NAND dengan dua *input* (X1, X2) dan satu *output* (Y). Dari ketiga gerbang logika dasar tersebut dapat dibangun dengan menggunakan arsitektur *neural network*. Pada setiap masing-masing arsitektur *neural network* mempunyai jumlah *input* dan *output* yang sama, dikarenakan mempunyai jumlah *input* dan *output* sama, hanya yang membedakan pada setiap bobotnya saja, maka dapat dibangun ke dalam satu arsitektur dengan hanya menggunakan pemilihan atas bobot Wa pada *line* 1 dan bobot Wb pada *Line* 2 maka arsitektur *neural network* dapat berfungsi sebagai fungsi gerbang logika yang diinginkan.

Bobot-bobot yang digunakan dalam proses pemilihan pada setiap *line* 1 dan *line* 2 perlu dicari pada setiap arsitekturnya. Proses pencarian bobot dilakukan dengan menggunakan algoritma program dilakukan melalui proses pelatihan. Selama pelatihan, bobot neuron, seperti (W1) dan (W2) untuk gerbang logika AND, (W3) dan (W4) untuk gerbang OR, serta (W5) dan (W6) untuk gerbang NAND. Proses seleksi dengan bobot proses ini bertujuan untuk menyesuaikan bobot sehingga neuron dapat menghasilkan *output* yang sesuai dengan fungsi logika masing-masing. Setelah bobot didapatkan, maka disimpan dalam *database* untuk digunakan kembali dalam implementasi jaringan *neural*.

Selanjutnya bobot yang sudah didapatkan akan dimasukan ke dalam algoritma program untuk proses seleksi pada setiap gerbang logika. Bobot-bobot yang berada di dalam suatu *database* akan dipanggil oleh sistem. Selama proses pemanggilan antara gerbang logika AND, OR, dan NAND, bobot yang digunakan merupakan bobot sesuai dengan gerbang logika yang diinginkan. Pemilihan dilakukan dengan cara menentukan gerbang logika yang diinginkan, maka sistem akan mengambil bobot yang sudah tersimpan untuk dimasukan pada Wa pada *line* 1 dan Wb pada *line* 2, kemudian menerapkannya pada arsitektur *neural network*. Dengan demikian, jaringan *neural* ini mampu beradaptasi untuk menjalankan berbagai fungsi logika hanya dengan mengubah bobotnya dengan catatan *input* dan *output* harus sama, sehingga menghasilkan *output* yang sesuai dengan gerbang logika yang dipilih.

3.3. Proses Penentuan Bobot Pada Rangkaian Kombinasional

Pada penelitian ini fungsi aktivasi *sigmoid* dan metode *backpropagation* digunakan dalam proses pencarian bobot pada setiap arsitektur *neural network*. Fungsi aktivasi *sigmoid* adalah fungsi aktivasi yang sering digunakan dalam jaringan syaraf tiruan (*neural networks*), terutama pada jaringan syaraf *forward* dan dalam pelatihan menggunakan algoritma *backpropagation*. Fungsi *sigmoid* mengatur *output* neuron dalam rentang antara 0 dan 1, yang mempermudah jaringan syaraf untuk mengklasifikasikan keputusan biner seperti operasi logika AND, OR, dan lainnya. Berikut adalah tahapan implementasinya:

1. Fungsi aktifasi *sigmoid*

   $\sigma(z) = \frac{1}{1+e^{-z}}$ (3.1)

   Dimana z adalah kombinasi linier dari *input*,bobot dan *Bias* : $z = X1.w1 + X2.w2 + b$

   W adalah bobot

   X adalah *input* neuron

   B adalah *Bias*

2. Turunan dari *sigmoid*

   $\frac{\partial Y_{pred}}{\partial z} = Y_{pred}.(1 - Y_{pred})$ (3.2)

3. Pelatihan *Backpropagation*

- F η = *Learning rate*
- *forward Propagation*:

  Menggunakan *Sigmoid*

  $y = \sigma(w.x + b$ (3.3)
- Hitung *Error* (*Mean Squared Error*)

  $E = \frac{1}{2}x(Y_{Target} - Y_{pred})^2$ (3.4)

Dimana Y_{Target} adalah target *output* dan Y_{pred} adalah *output* prediksi dari fungsi *sigmoid*

- Pembaruan Bobot dan *Bias*

  Bobot= $b_{baru} = b_{lama-}\eta.\frac{\partial E}{\partial w}$ (3.5)

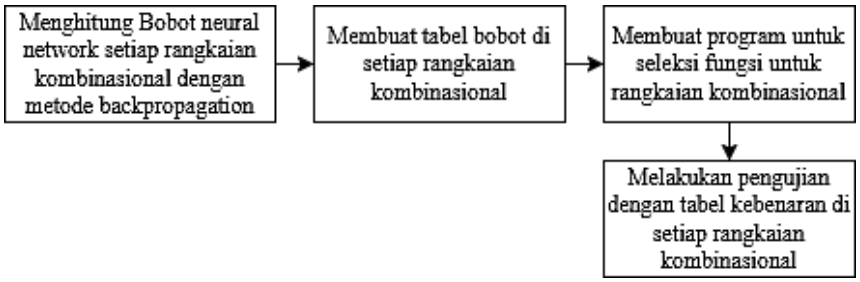
  Bias = $b_{baru} = b_{lama-}\eta.\frac{\partial E}{\partial b}$ (3.6)

Keterangan :

- $\frac{\partial E}{\partial b}$ = gradien dari fungsi *error* terhadap *bias*
- $\frac{\partial E}{\partial w}$ = gradien dari fungsi *error* terhadap bobot

3.4. Diagram Blok Konsep Perancangan Penelitian

Adapun tahapan yang dilakukan pada penelitian ini ditunjukkan pada Gambar 3.2 merupakan diagram blok konsep pada perancangan penelitian.



Gambar 3.2. Diagram Blok Konsep Perancangan

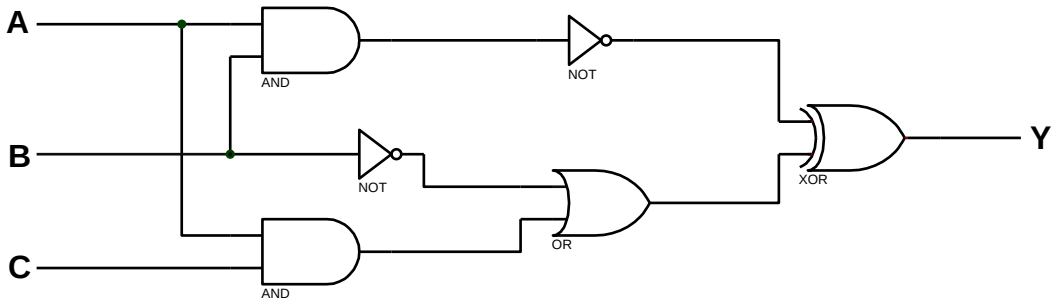
Diagram pada Gambar 3.2 merupakan alur proses tahap penelitian, tahap pertama dalam proses ini adalah menghitung bobot untuk *neural network* pada setiap rangkaian logika kombinasional menggunakan metode *backpropagation*. Metode ini memungkinkan jaringan *neural* untuk mengurangi kesalahan pada *output* dengan menyesuaikan bobot secara bertahap hingga diperoleh bobot optimal yang sesuai dengan fungsi logika yang dituju. Setelah perhitungan bobot selesai, tabel bobot disusun untuk setiap rangkaian kombinasional. Tabel ini berfungsi sebagai panduan bobot dalam program agar bisa dipanggil sesuai kebutuhan.

Langkah berikutnya adalah mengembangkan program yang memungkinkan pengguna memilih fungsi logika yang diinginkan untuk rangkaian kombinasional. Program ini dirancang agar dapat mengambil bobot dari tabel yang telah disesuaikan dengan pilihan fungsi logika pengguna. Dengan bobot yang tepat, jaringan *neural* diatur untuk beroperasi sesuai *input* yang diberikan, memungkinkan jaringan menghasilkan *output* yang sesuai fungsi logika yang dipilih.

Setelah konfigurasi bobot diterapkan, langkah terakhir adalah melakukan pengujian untuk memastikan jaringan *neural* dapat menghasilkan *output* yang sesuai dengan fungsi logika yang telah dipilih. Pengujian ini menggunakan tabel kebenaran sebagai acuan untuk membandingkan *output* jaringan dengan nilai yang diharapkan. Hal ini bertujuan mengevaluasi apakah *output* jaringan *neural* telah sesuai, sehingga memastikan jaringan dapat berfungsi dengan tingkat akurasi tinggi dalam menggantikan peran rangkaian logika kombinasional.

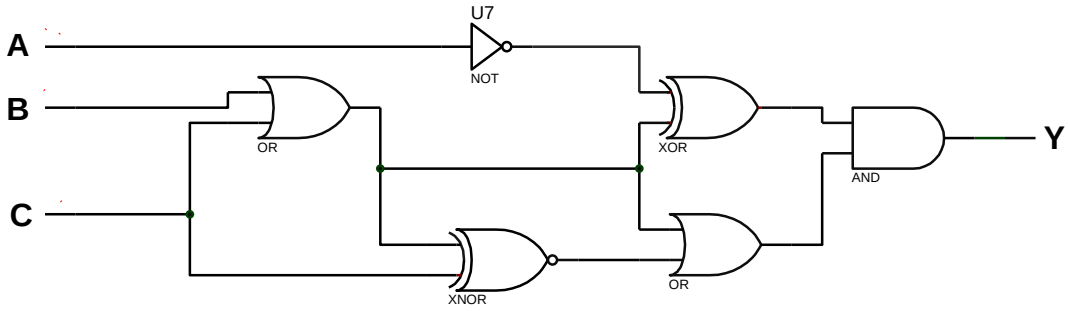
3.5. Implementasi *Neural Network* Pada Sistem Kombinasional

Berdasarkan konsep sederhana yang sudah dirancang seperti pada Gambar 3.1 dengan mengimplementasikan berbagai fungsi gerbang logika dasar ke dalam arsitektur *neural network.* Konsep ini menyatakan bahwa dengan menggunakan satu arsitektur *neural network* saja dapat mengintegrasikan berbagai fungsi gerbang logika. Pada tahap ini, implementasi dilakukan dengan mengintegrasikan berbagai fungsi rangkaian kombinasional ke dalam sebuah arsitektur *neural network*. Sistem ini dirancang untuk memproses berbagai fungsi rangkaian kombinasional ke dalam satu arsitektur *neural network* dengan catataan *input* dan *output* sama disetiap arsitekturnya. Dengan memanfaatkan bobot yang didapat pada proses pelatihan jaringan *neural network* yang akan disimpan dan dimasukan ke dalam coding seleksi rangkaian kombinasional, maka pada setiap rangkaian kombinasional dapat diaktifkan sesuai dengan kebutuhan dengan hanya mengubah bobot pada arsitekturnya. Pada Gambar 3.3, Gambar 3.4, Gambar 3.5, Gambar 3.6 dan Gambar 3.7 merupakan beberapa rangkaian kombinasional yang berbeda dengan jumlah *input* dan *output* yang sama, yang akan diimplementasikan ke dalam arsitektur *neural network*.



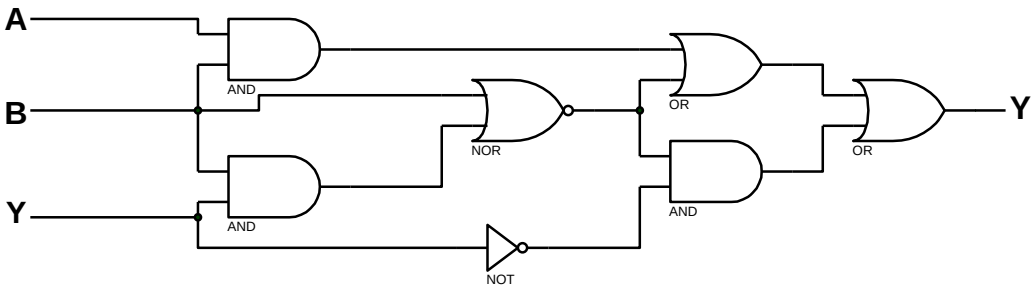
Gambar 3.3. Rangkaian Kombinasional 1

$$Y = (\overline{A.B}) \oplus \left(\bar{B} + (A.C)\right) \quad (3.7)$$



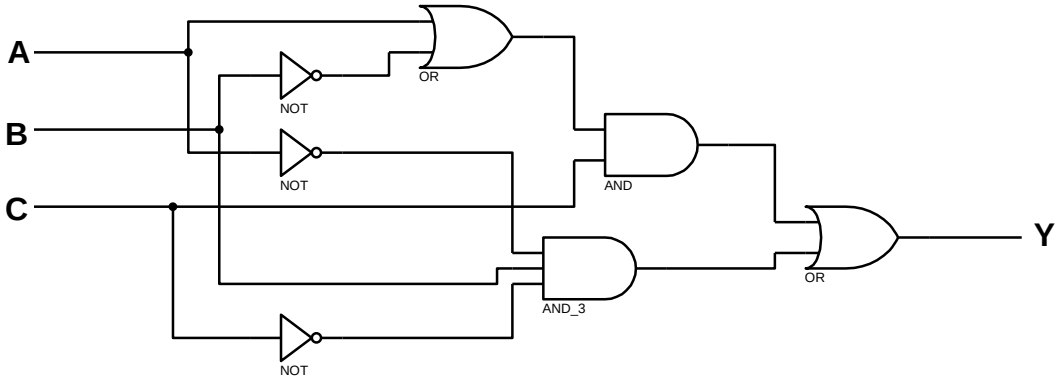
Gambar 3.4. Rangkaian Kombinasional 2

$$Y = \left(\bar{A} \oplus (B + C)\right).\left((B + C) + \overline{(\mathrm{B} + \mathrm{C}) \oplus \mathrm{C}}\right) \quad (3.8)$$



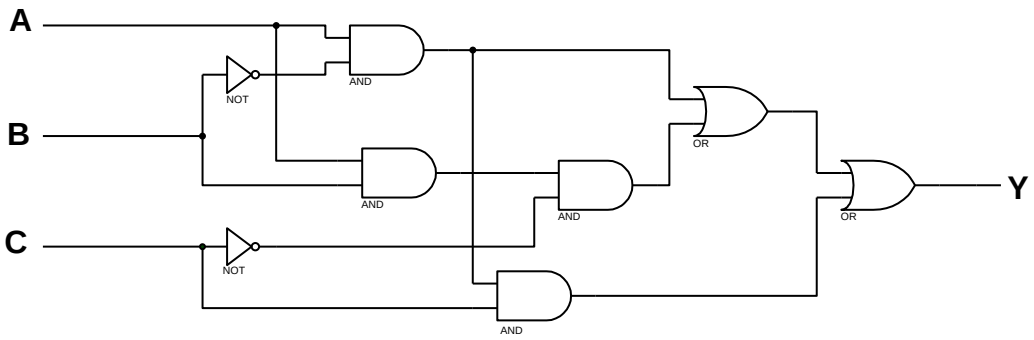
Gambar 3.5. Rangkaian Kombinasional 3

$$Y = \left((A.B) + \left(\overline{B + (B.C)}\right)\right) + \left((\overline{\mathrm{B} + (\mathrm{B}.\mathrm{C})}.\bar{\mathrm{C}}\right) \quad (3.9)$$



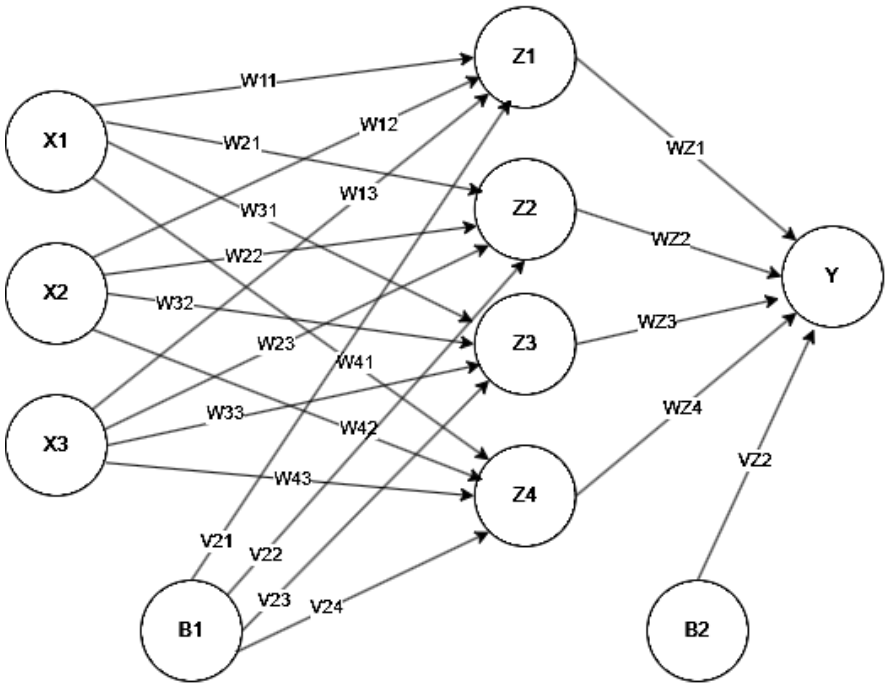
Gambar 3.6. Rangkaian Kombinasional 4

$$Y = \left((A + \bar{B}).C\right) + (\bar{A}.B.\bar{C}) \quad (3.1$$

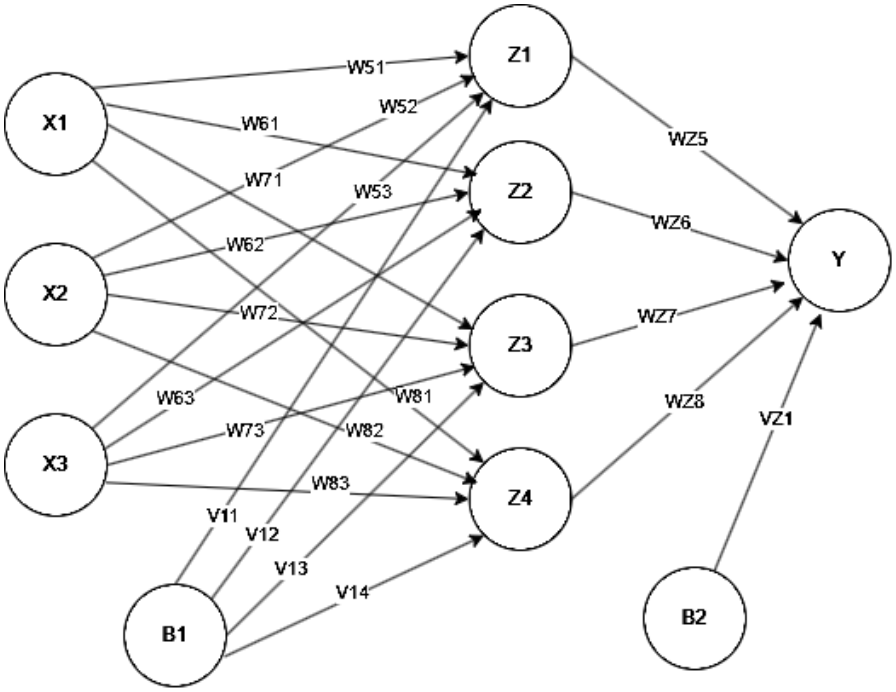


Gambar 3.7. Rangkaian Kombinasional 5

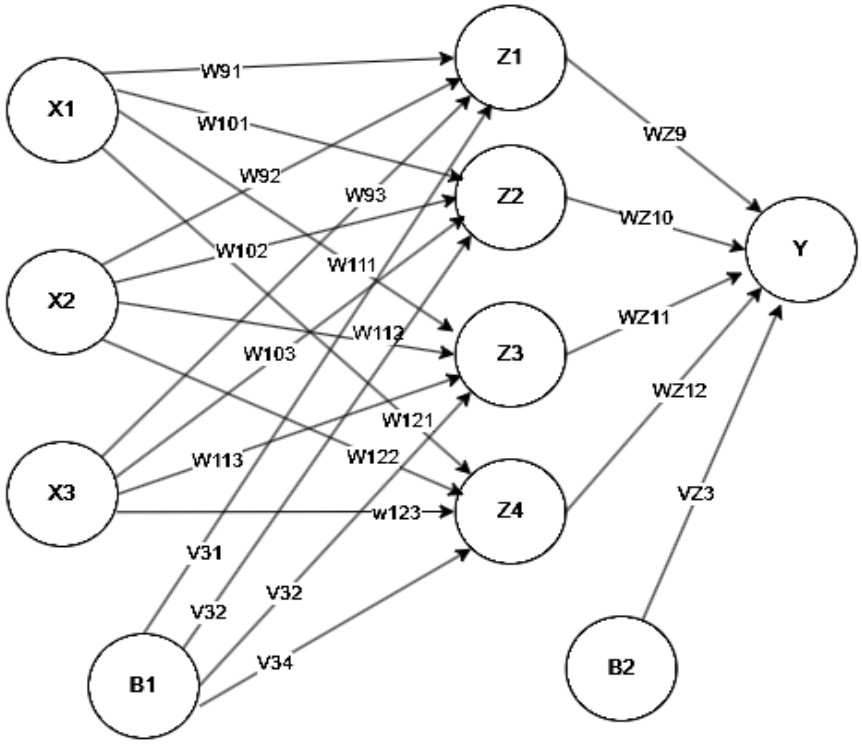
$$Y = \left((A.\overline{B}) + (A.B.\overline{C})\right) + (A.\overline{B}.C) \quad (3.11)$$



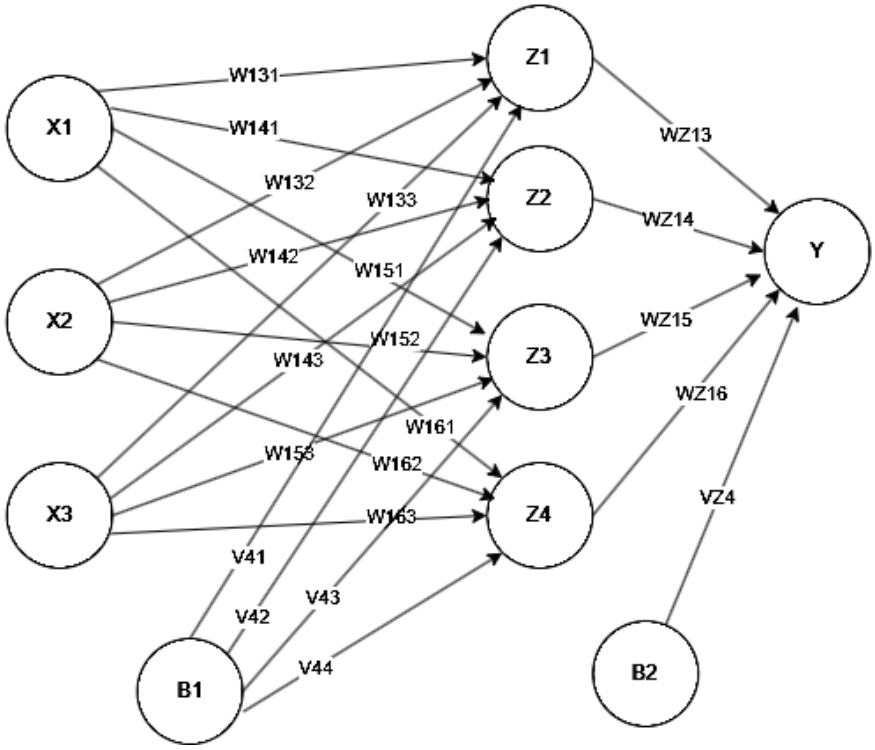
Gambar 3.8. Arsitektur *Neural Network* Kombinasional 1



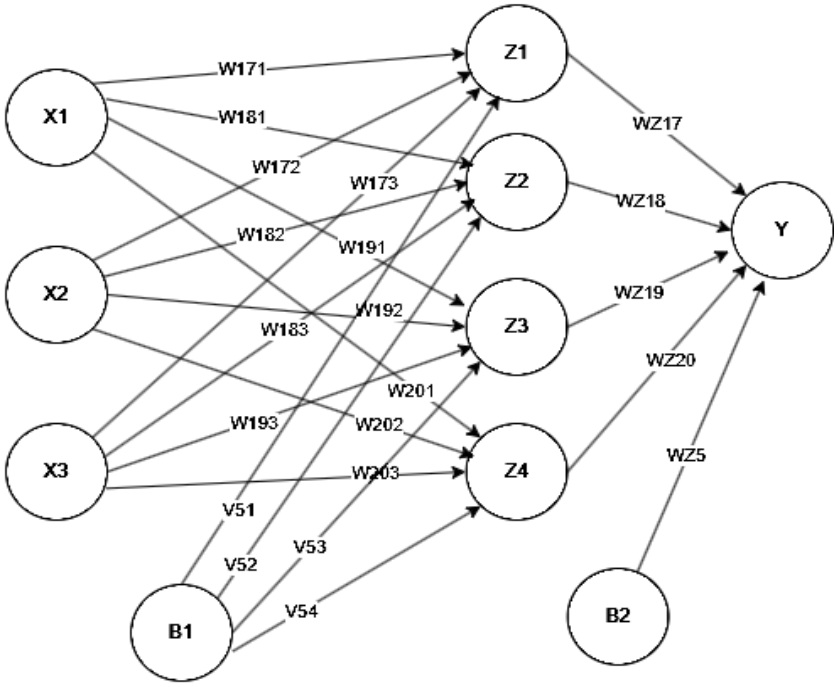
Gambar 3.9. Arsitektur *Neural Network* Kombinasional 2



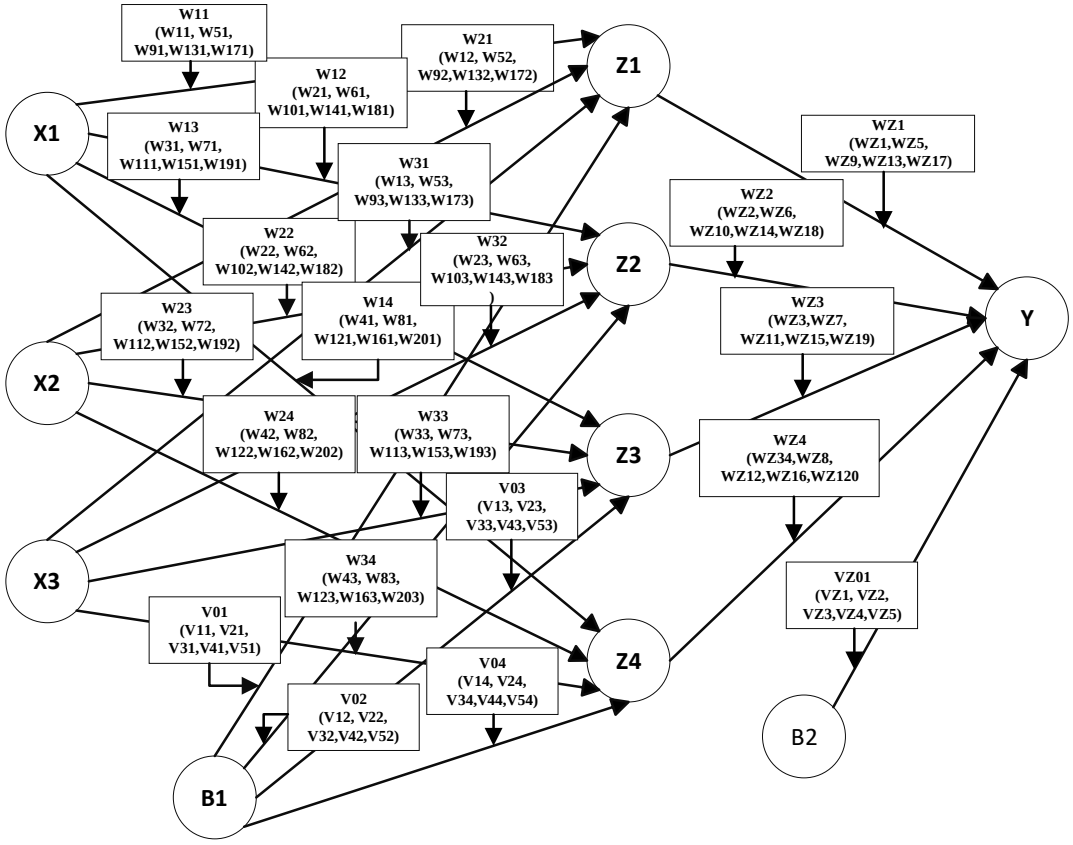
Gambar 3.10. Arsitektur *Neural Network* Kombinasional 3



Gambar 3.11. Arsitektur *Neural Network* Kombinasional 4



Gambar 3.12. Arsitektur *Neural Network* Kombinasional 5



Gambar 3.13. Implementasi Lima Arsitektur Ke Dalam Satu Arsitektur *Neural Network*

Berdasarkan Gambar 3.3, Gambar 3.4, Gambar 3.5, Gambar 3.6, dan Gambar 3.7 merupakan lima rangkaian kombinasional berbeda dengan masing-masing *input* dan *output* yang sama. Dari kelima fungsi rangkaian kombinasional tersebut dapat diimplementasikan ke dalam arsitektur *neural network* seperti pada Gambar 3.8, Gambar 3.9, Gambar 3.10, Gambar 3.11 dan Gambar 3.12 dengan menggunakan arsitektur *multi-layer perceptron* dengan tiga *input layer* (X1, X2, X3), empat neuron *hidden layer* (Z1, Z2, Z3, Z4), dan satu neuron *output layer* (Y).

Selanjutnya, dalam setiap arsitektur *neural network* tersebut, bobot dan *bias* yang sesuai perlu dicari dan ditentukan melalui proses pelatihan. Bobot-bobot dan *bias* ini nantinya akan digunakan untuk proses pemilihan pada setiap kombinasional fungsi logika yang diinginkan, sehingga arsitektur *neural network* dapat beradaptasi untuk menghasilkan *output* yang sesuai dengan fungsi logika masing-masing.

Proses pembelajaran dimulai dengan langkah *feedforward*, dimana sinyal *input* diberi bobot dan *bias*, kemudian melewati lapisan-lapisan *neural network* untuk menghasilkan *output*. *Output* ini kemudian dibandingkan dengan target yang diharapkan, dan selisihnya dihitung sebagai *error*. *Error* ini digunakan dalam proses *backpropagation*, dimana bobot dan *bias* diperbarui secara iteratif untuk meminimalkan *error*.

Pada setiap neuron, selain bobot yang menghubungkan neuron-neuron antar lapisan, juga terdapat nilai *bias* yang bertujuan untuk menggeser aktivasi neuron agar jaringan dapat belajar pola lebih kompleks dan meningkatkan akurasi hasilnya. Setiap rangkaian kombinasional memiliki arsitektur *neural network* yang sama, yang membedakan hanya bobot dan *bias* pada setiap arsitekturnya. Karena setiap rangkaian kombinasional memiliki arsitektur yang sama, maka seluruh rangkaian tersebut dapat dibangun dalam satu arsitektur *neural network* seperti yang ditunjukkan pada Gambar 3.13, dengan catatan *input*, *output*, serta nilai *bias* tetap konsisten. Penggunaan satu arsitektur dengan jumlah input dan output adalah sama hanya yang membedakan pada bobotnya, bobot-botot disetiap arsitektur nantinya digunakan dalam proses seleksi dengan algoritma program disetiap rangkaian kombinasional yang akan diaktifkan. Pada Tabel 3.1 dan Tabel 3.2 adalah tabel bobot pada stiap arsitektur *neural network* yang didapatkan selama pelatihan

Tabel 3.1. Bobot Antara *Input Layer* (X) Dan *Hidden Layer* (Z)

Hubungan	Arsitektur 1	Arsitektur 2	Arsitektur 3	Arsitektur 4	Arsitektur 5
Neuron	Bobot	Bobot	Bobot	Bobot	Bobot
X1, Z1	W11	W51	W91	W131	W171
X1, Z2	W21	W61	W101	W141	W181
X1, Z3	W31	W71	W111	W151	W191
X1, Z4	W41	W81	W121	W161	W201
X2, Z1	W12	W52	W92	W132	W172
X2, Z2	W22	W62	W102	W142	W182
X2, Z3	W32	W72	W112	W152	W192
X2, Z4	W42	W82	W122	W162	W202
X3, Z1	W13	W53	W93	W133	W173
X3, Z2	W23	W63	W103	W143	W183
X3, Z3	W33	W73	W113	W153	W193
X3, Z4	W43	W83	W123	W164	W203

Tabel 3.2. *Bias Hidden Layer*

Hubungan *Bias*	*Bias* Arsitektur 1	*Bias* Arsitektur 2	*Bias* Arsitektur 3	*Bias* Arsitektur 4	*Bias* Arsitektur5
B1, Z1	V11	V21	V31	V41	V51
B1, Z2	V12	V22	V32	V42	V52
B1, Z3	V13	V23	V33	V43	V53
B1, Z4	V14	V24	V34	V44	V54

Pada Tabel 3.2 menunjukkan bobot dan *bias* antara *input layer* (X) dan *hidden layer* (Z) pada setiap arsitektur *neural network* yang sudah dilatih dan disimpan secara terpisah. Pada Tabel 3.2 menunjukkan nilai bobot antara *input* dan *hidden layer* dan Tabel 3.2 menunjukkan nilai bobot *bias.* Pada kondisi memilih pada rangkaian kombinasional 1, maka sistem akan mengambil bobot yang sesuai pada arsitektur 1 serta nilai *bias* pada kolom *bias* arsitektur 1. Sistem kemudian akan mengatur bobot dan *bias* ini untuk memastikan *output* yang dihasilkan sesuai dengan target yang telah ditetapkan. Demikian juga untuk kombinasional logika lainnya, seperti kombinasional 2, kombinasional 3, kombinasional 4, dan kombinasional 5, sistem akan menggunakan bobot dan *bias* yang sesuai dari masing-masing arsitektur untuk mencapai target yang diinginkan. Selanjutnya untuk bobot antara *hidden layer* (Z) dan *output layer* (Y) serta bobot *bias output* ditunjukkan pada Tabel 3.3 dan Tabel 3.4

Tabel 3.3. Bobot Antara *Hidden Layer* (Z) Dan *Output Layer*

Hubungan	Arsitektur 1	Arsitektur 2	Arsitektur 3	Arsitektur 4	Arsitektur 5
Neuron	Bobot	Bobot	Bobot	Bobot	Bobot
Z1, Y	WZ1	WZ5	WZ9	WZ13	WZ17
Z2, Y	WZ2	WZ6	WZ10	WZ14	WZ18
Z3, Y	WZ3	WZ7	WZ11	WZ15	WZ19
Z4, Y	WZ4	WZ8	WZ12	WZ16	WZ20

Tabel 3.4. Bobot *Bias Output*

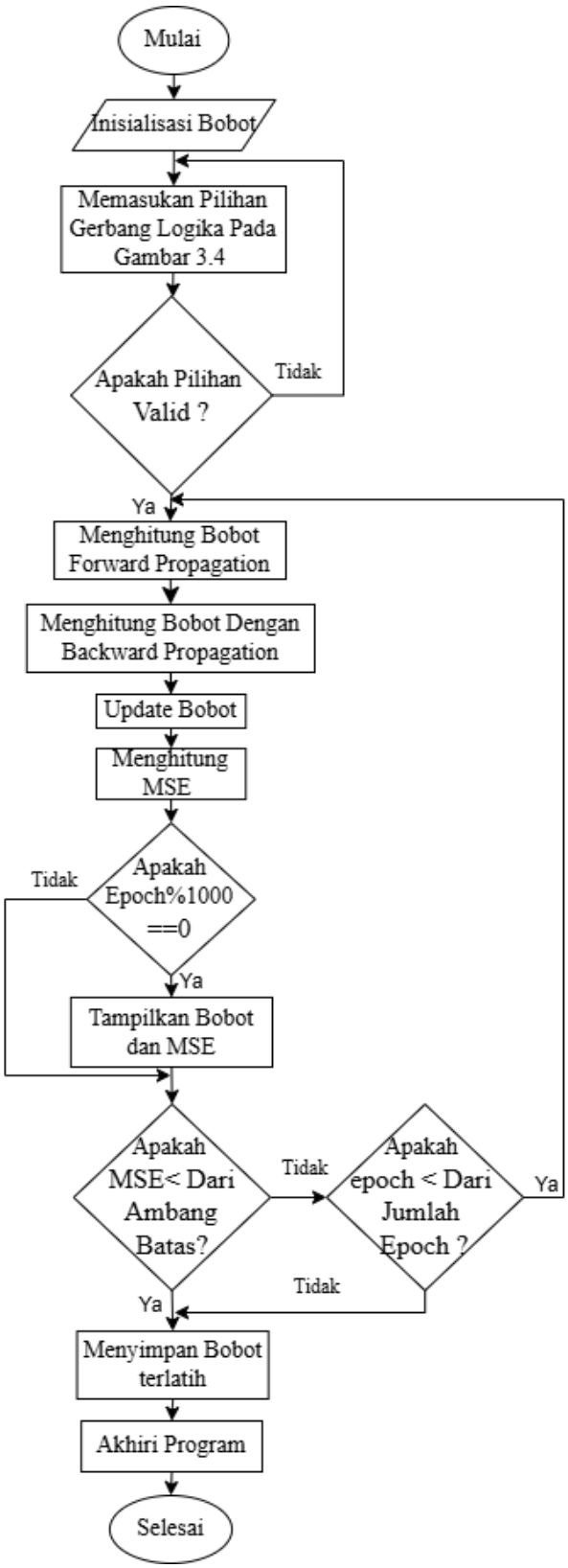
Hubungan *Bias*	*Bias* Arsitektur 1	*Bias* Arsitektur 2	*Bias* Arsitektur 3	*Bias* Arsitektur 4	*Bias* Arsitektur 5
B2, Y	VZ1	VZ2	VZ3	VZ4	VZ5

Pada Tabel 3.3 dan Tabel 3.4 menunjukkan bobot antara neuron *hidden layer* (Z) ke *otput layer* (Y) dan bobot *bias output*, untuk setiap arsitektur *neural network* yang telah dilatih dan disimpan secara terpisah. Ketika sistem memilih kondisi pada rangkaian kombinasional 1, bobot dan *bias* yang digunakan antara neuron *hidden layer* (Z) dan *output layer* (Y) adalah bobot dari arsitektur 1, yaitu (WZ1, WZ2, WZ3, WZ4), serta *bias* pada rangkaian 1 (VZ1). Nilai bobot dan *bias* ini kemudian disesuaikan untuk mencapai target *output* yang diinginkan. Begitu pula untuk kombinasional 2, kombinasional 3, kombinasional 4 dan kombinasional 5 sistem akan memilih bobot yang sesuai dengan masing-masing kombinasi tersebut. Dengan demikian hanya memasukkan kedua set bobot dan *bias*, yaitu bobot dan *bias* antara *input layer* X dan *hidden layer* Z dari Tabel 3.1 dan Tabel 3.4, serta bobot dan *bias* antara neuron *hidden layer* (Z) ke *otput layer* (Y) dari Tabel 3.3 dan 3.4 jaringan syaraf tiruan (JST) dapat berfungsi sebagai berbagai rangkaian logika kombinasional. Setiap kali sistem memilih satu kombinasional logika, sistem akan mengambil bobot dan *bias* dari keempat tabel sesuai kombinasional yang dipilih untuk menghasilkan *output* yang diinginkan. Dengan hanya mengganti bobot terlatih dengan catatan *input* dan *output* sama, sistem dapat menjalankan berbagai fungsi logika kombinasional.

3.6. Diagram Alir Perancangan Sistem Algoritma Program

Adapun diagram alir perancangan sistem algoritma program pada pencarian bobot dan sistem seleksi rangkaian pengujian pada penelitian yang akan dilaksanakan dan dijelaskan dalam bentuk diagram alir yang ditunjukkan pada Gambar 3.14 dan Gambar 3.15.

3.6.1. Proses Pelatihan Bobot Setiap Kombinasional



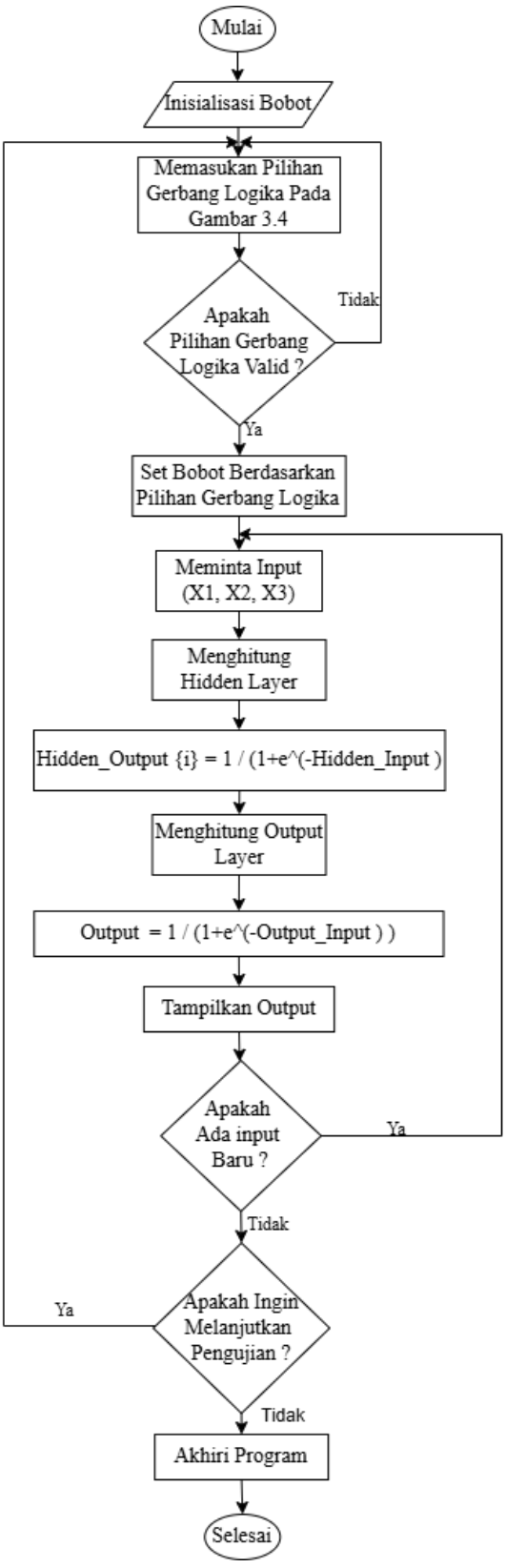
Gambar 3.14. Diagram Alir Sistem Pelatihan Bobot Perancangan Kombinasional

Proses pelatihan *neural network* untuk mensimulasikan gerbang logika dimulai dengan inisialisasi bobot jaringan. Setelah inisialisasi dilakukan, pengguna diminta untuk memasukkan pilihan gerbang logika yang akan digunakan, seperti kombinasional 1, kombinasional 2 dan sebagainya. Sistem kemudian memvalidasi *input* pengguna. Jika *input* tersebut valid, bobot *neural network* diatur sesuai dengan pilihan gerbang logika yang dipilih.

Tahap berikutnya adalah permintaan *input* data latih, dimana data ini akan digunakan dalam proses pelatihan *neural network*. Pelatihan dimulai dengan *forward propagation*, yaitu perhitungan *output* jaringan berdasarkan *input* yang diberikan. Setelah *output* dihitung, dilakukan *backward propagation* untuk menghitung *error* dan memperbarui bobot jaringan agar *error* tersebut diminimalkan.

Selanjutnya, nilai *Mean Squared Error* (MSE) dihitung untuk mengevaluasi performa jaringan terhadap target yang diinginkan. Setiap kelipatan 100 *epoch*, sistem akan menampilkan bobot dan nilai MSE untuk memantau perkembangan pelatihan. Jika nilai MSE sudah lebih kecil dari ambang batas yang ditentukan, pelatihan dihentikan, dan bobot yang telah dilatih disimpan untuk digunakan pada tahap berikutnya. Jika nilai MSE masih di atas ambang batas, pelatihan dilanjutkan hingga mencapai jumlah *epoch* yang ditentukan. Setelah jumlah *epoch* terpenuhi atau MSE berada di bawah ambang batas, pelatihan dianggap selesai dan program berakhir.

3.6.2. Proses Pengujian Seleksi Rangkaian Kombinasional



Gambar 3.15. Diagram Alir *Neural Network* Proses Seleksi Dengan Bobot

Proses seleksi setiap bobot yang didapat untuk fungsi rangkaian kombinasion pada model jaringan syaraf tiruan (JST) untuk mensimulasikan fungsi gerbang logika dimulai dengan langkah inisialisasi bobot. Setelah bobot diinisialisasi, pengguna diminta untuk memasukkan pilihan gerbang logika yang diinginkan. Selanjutnya, sistem melakukan verifikasi terhadap validasi pilihan gerbang logika tersebut. Jika pilihan yang dimasukkan tidak valid, sistem akan menampilkan pesan kesalahan dan mengulangi *input* yang akan dimasukan. Sebaliknya, jika pilihan valid, sistem akan melanjutkan dengan mengatur bobot berdasarkan gerbang logika yang dipilih. Setelah bobot ditetapkan, sistem meminta pengguna untuk memasukkan nilai *input*, yakni X1, X2, dan X3. Nilai *input* ini kemudian digunakan untuk menghitung *output* pada *hidden layer* dengan menggunakan persamaan aktivasi *sigmoid*, yaitu:

$$Hidden\_Layer = \frac{1}{1+e^{-Hiden\_Input}} \quad (3.7)$$

Setelah lapisan tersembunyi dihitung, sistem akan melanjutkan ke perhitungan *output layer* dengan menggunakan persamaan yang sama, yaitu:

$$Output\_Layer = \frac{1}{1+e^{-Output\_Input}} \quad (3.8)$$

Hasil *output* dari perhitungan *output layer* kemudian ditampilkan kepada pengguna. Setelah hasil *output* ditampilkan, sistem akan mengevaluasi apakah pengguna ingin memasukkan *input* baru atau melanjutkan pengujian dengan data yang ada. Jika tidak ada *input* baru dan pengguna tidak ingin melanjutkan pengujian, maka program dapat diakhiri.

3.7. Alat Dan Bahan

Adapun alat dan bahan yang digunakan pada penelitian yang dilakukan ditunjukkan pada Tabel 3.5.

Tabel 3. 5 Alat Dan Bahan Penelitian

Alat dan Bahan	Justifikasi Penggunaan
Laptop Asus, *AMD Ryzen 3 7320U with Radeon Graphics 2.40 GHz, Windows 11 Home Single Language, RAM 8,00 GB 64-bit operating system, x64-based processor*	Sebagai tempat menyusun algoritma, pelaksanaan simulasi, serta penyusunan proposal.
Software Python	Bahasa pemograman yang digunakan untuk membuat coding untuk penelitian.
Sofwrae proteus	Digunakan untuk membuat rangkaian kombinasional dan membantu proses melihat hasil rangkaian gerbang logika.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Adapun kesimpulan dari penelitian ini adalah sebagai berikut:

1. Sebuah arsitektur yang diusulkan berhasil memodelkan lima rangkaian kombinasional dengan *database* bobot *neural network.*
2. Algoritma program yang telah dibuat berhasil menjalankan tugas untuk menempatkan bobot pada jaringan neuron sesuai dengan rangkaian kombinasional yang diaktifkan.

5.2. Saran

Adapun saran dari penelitian ini adalah sebagai berikut:

1. Pada penelitian ini agar tidak hanya menggunakan rangkaian kombinasional dengan tiga *input* dan satu *output*, tetapi juga mempertimbangkan penggunaan rangkaian kombinasional dengan jumlah *input* dan *output* yang lebih banyak. Selain itu, pada sistem jaringan syaraf tiruan atau arsitektur *neural network*, disarankan untuk menggunakan konfigurasi *input* dan *output* yang fleksibel sehingga dapat disesuaikan dengan karakteristik rangkaian kombinasional yang digunakan.
2. Pada penelitian ini, digunakan dua *code* terpisah untuk tahap pencarian bobot dan tahap seleksi. Untuk penelitian selanjutnya, disarankan agar bobot yang diperoleh disimpan dalam *database*, sehingga proses pemanggilan bobot dapat dilakukan langsung dari *database* tersebut, memungkinkan proses seleksi dilakukan dengan menggunakan satu *code* saja.

DAFTAR PUSTAKA

Aziz, A.S. (2022) 'Model Neuron *Mc Culloch-Pitts* dalam Pengenalan Pola Logika Dasar', *JEECOM Journal of Electrical Engineering and Computer*, 4(2), pp. 51–56. Available at: https://doi.org/10.33650/jeecom.v4i2.3673.

Denny Pratama (2024) 'Generator Set Uji Untuk Diagnosis Kesalahan Rangkaian Kombinasional Multiplexer 4 *Line To 1 Line* Menggunakan Metode *Artificial Neural Network*', *(Tesis)* Fakultas Teknik. Teknik Elektro. Universitas Lampung.

Hadimarta, T.F., Muhima, R.R. and Kurniawan, M. (2020) 'Implementasi *Multilayer* Perceptron Pada Jaringan Saraf Tiruan Untuk Memprediksi Nilai Valuta Asing', *INTEGER: Journal of Information Technology*, 5(1), pp. 56–63. Available at: https://doi.org/10.31284/j.integer.2020.v5i1.909.

Hasibuan, P. *et al.* (2024) 'Implementasi Penggunaan Aplikasi Meeting Zoom Dalam Pembelajaran Matematika Pada Materi Barisan', *Mathematical and Data Analytics*, 1(1), pp. 31–37. Available at: https://doi.org/10.47709/mda.v1i1.3887.

Indrawaty, Y., Kristina, L. and Nugraha, S. (2012) 'Aplikasi Pembelajaran Rangkaian Kombinasional dengan Menggunakan Skenario Multimedia Interaktif Model *Timeline* Tree', *Jurnal Informatika*, 3(2), p. 11. Available at: https://lib.itenas.ac.id/kti.

Jan, M., Sianipar, R.H. and Sultan (2015) 'Perancangan Simulator Rangkaian Logika Dengan Visual C ++', *Dielektrika*, 2(2), p. 156. Available at: https://dielektrika.unram.ac.id/index.php/dielektrika/article/view/71.

Pamungkas, I., Sumadi, S. and Alam, S. (2022) 'Studi Komparasi Fungsi Aktivasi *Sigmoid* Biner, *Sigmoid* Bipolar dan *Linear* pada Jaringan Saraf Tiruan dalam Menentukan Warna RGB Menggunakan Matlab', *Jurnal Serambi Engineering*, 7(4), pp. 2–10. Available at: https://doi.org/10.32672/jse.v7i4.4776.

Paramita, S.P. (2010) *Model Backpropagation Neural Network* Untuk Peramalan Kasus Demam Berdarah di *D.I YOGYAKARTA*. Universitas Negeri Yogyakarta.

Parinduri, I. and Nurhabibah Hutagalung, S. (2019) 'Perangkaian Gerbang Logika Dengan Menggunakan Matlab (Simulink)', *JURTEKSI (Jurnal Teknologi dan Sistem Informasi)*, 5(1), pp. 63–70. Available at: https://doi.org/10.33330/jurteksi.v5i1.300.

Rachman, A.S., Cholissodin, I. and Fauzi, M.A. (2018) 'Peramalan Produksi Gula Menggunakan Metode Jaringan Syaraf Tiruan *Backpropagation* Pada PG Candi Baru Sidoarjo', *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 2(4), pp. 1683–1689. Available at: https://j-ptiik.ub.ac.id/.

Rahman, S. *et al.* (2023) *Python :* Dasar Dan Pemrograman Berorientasi Objek, Penerbit Tahta Media. Available at: https://www.tahtamedia.com/.

Raji, M., Sabet, M.A. and Ghavami, B. (2019) '*Soft Error Reliability Improvement of Digital Circuits by Exploiting a Fast Gate Sizing Scheme*', *IEEE Access*, 7, pp. 66485–66495. Available at: https://doi.org/10.1109/ACCESS.2019.2902505.

Salsabila, B. and Cholissodin, I. (2020) 'Prediksi Permintaan Keripik Buah dengan Metode Jaringan Syaraf Tiruan *Backpropagation* (Studi Kasus: CV. Arjuna 999)', 4(6), pp. 1667–1674. Available at: http://j-ptiik.ub.ac.id.

Sugiartowo and Ambo, S.N. (2018) 'Simulasi Rangkaian Kombinasional Sebagai Media Pembelajaran Sistem Digital Pada Fakultas Teknik Universitas Muhammadiyah Jakarta', *Umj*, 005, pp. 1–11. Available at: https://jurnal.umj.ac.id/index.php/semnastek/article/view/3407.

Sulthanah, R. and Ahmad, N. (2023) ‘Penerapan Metode *McCulloch-Pitts* Menggunakan Python Untuk Pengujian Pengenalan Pola Operator AND, Operator OR, Operator XOR Pada Fungsi Logika’, *J-Intech*, 11(2), pp. 347–360. Available at: https://doi.org/10.32664/j-intech.v11i2.1022.

Suryadibrata, A. and Chandra, D.P. (2020) ‘Implementasi Jaringan Saraf Tiruan *Backpropagation* untuk Pengenalan Karakter pada Dokumen Tercetak’, *Ultima Computing : Jurnal Sistem Komputer*, 11(2), pp. 81–89. Available at: https://doi.org/10.31937/sk.v11i2.1456.

Wahyono, E.B. (2011) ‘Penerapan Jaringan Syaraf Tiruan Metode Propagasi Balik Dalam Mengenali Aksara Jawa’, *Jurnal Sains & Teknologi*, 2(1), pp. 21–28. Available at: ttp://repository.unsada.ac.id/606/1.

Wuryandari, M.D. and Afrianto, I. (2012) ‘Perbandingan Metode Jaringan Syaraf Tiruan *Backpropagation* Dan *Learning Vector Quantization* Pada Pengenalan Wajah’, *Jurnal Komputer dan Informatika (Komputa)*, 1(1), pp. 45–51.