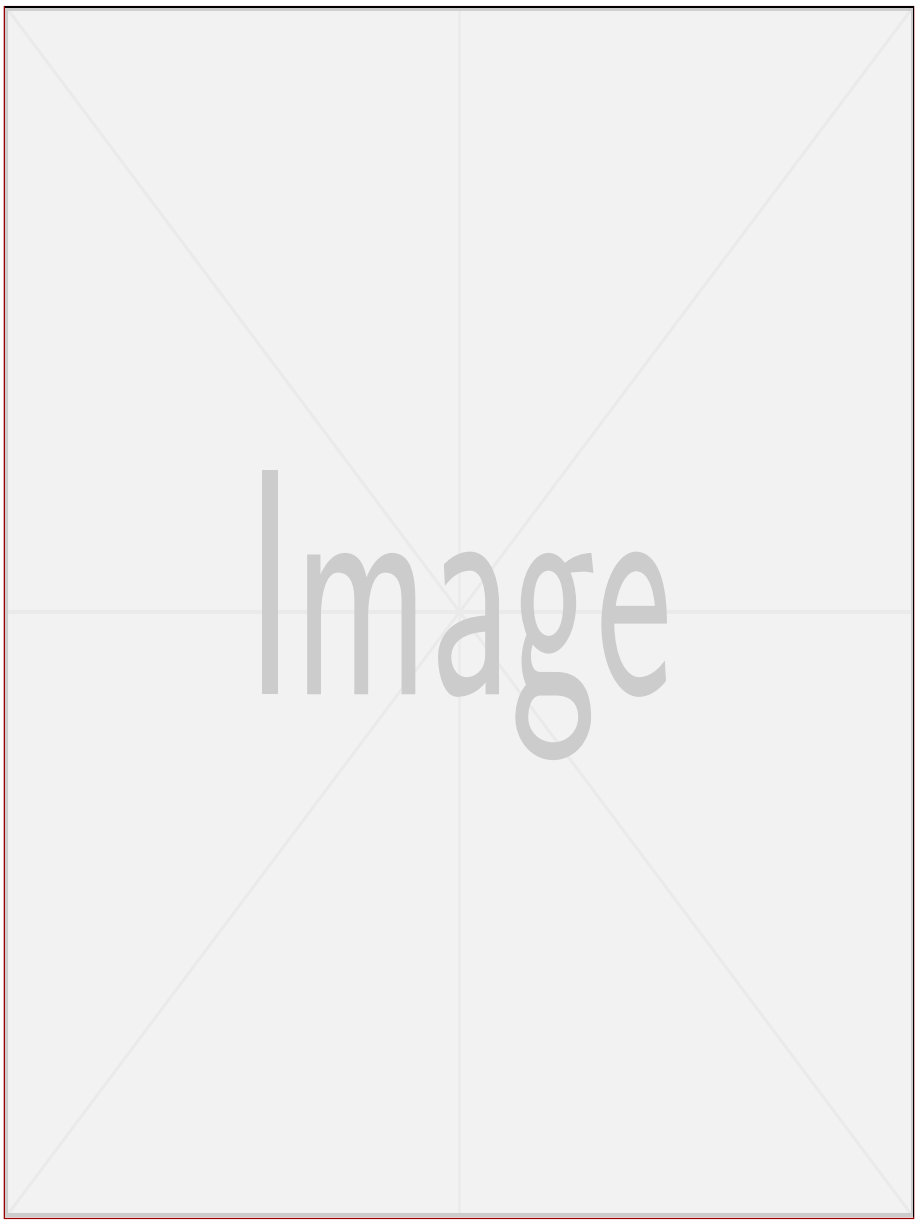
DEEP THINKING

Notes on Learning Machines and the Human Mind

Fauzy Madani

SMK Negeri 1 Garut

fauzy_madani@smknegeri1garut.sch.id



About the Author

Fauzy Madani is not just a student of technology, but a contemplator of knowledge. He believes that understanding machines helps us understand ourselves — and that clarity is the highest form of empathy.

As a passionate software developer, his fascination lies in the overlap between artificial intelligence and human intuition, where logic meets mystery.

> "To understand the machine is to reflect on the self."

CONTENTS

"The true logic of this world is in the calculus of probabilities."

— ***James C. Maxwell***

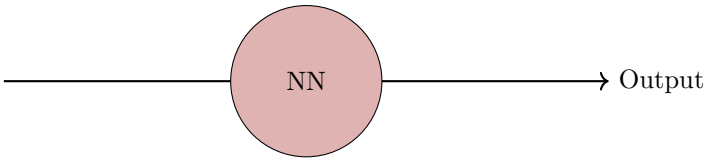
CHAPTER

1

HOW TO USE THIS BOOK

"The true logic of this world is in the calculus of probabilities."

— ***James C. Maxwell***



Visual representation of a simple neural network.

1.1 Introduction

First of all, involves carefully examining ideas beyond their surface level by systematically analyzing relationships, causes, and implications. It requires going beyond just accepting information as it is, instead questioning assumptions and exploring multiple layers of reasoning to understand the underlying principles. This kind of thinking uses formal logical processes like deduction and induction to draw well-founded conclusions and often involves breaking down complex problems into smaller, more manageable parts to analyze them thoroughly. From a technological standpoint, deep thinking means understanding how different parts of a system interact and affect each other, rather than looking at components in isolation. It involves creating solutions that are efficient, scalable, and robust by anticipating potential problems and edge cases. Technological deep thinking also requires a mindset of careful design — building abstractions and modular structures that manage complexity while maintaining flexibility. This means thinking not only about how something works right now but also how it will behave under different conditions or how it can be adapted in the future. Essentially, deep thinking in both realms is about moving beyond superficial understanding to grasp the intricate details and bigger picture simultaneously, enabling better decision-making and problem-solving.

1.1.1 What makes this book so valuable

Deciphering dead languages. Detecting malignant tumours. develop a skill that's crucial but often overlooked: the ability to analyze complex problems thoughtfully and thoroughly. In today's fast-paced world, many people tend to skim information or jump to quick conclusions, but deep thinking encourages slowing down, questioning assumptions, and understanding systems at a fundamental level. Such a book would provide practical frameworks and mental tools to approach challenges more effectively, whether in everyday life, science, engineering, or technology. It can also bridge the gap between abstract reasoning and real-world application, helping readers not just to think harder but to think smarter—anticipating problems, designing better solutions, and making decisions that stand the test of time. Ultimately, its value lies in empowering readers to cultivate clarity, insight, and creativity, which are essential for innovation and meaningful progress in any field.

CHAPTER

2

DEEP LEARNING BASICS

"The true logic of this world is in the calculus of probabilities."

— ***James C. Maxwell***

2.1 What is a Neural Network?

ARTIFICIAL type of computational model inspired by the way biological brains work, designed to recognize patterns and solve complex problems by learning from data. At its core, a neural network is composed of many interconnected units called neurons or nodes, arranged in layers. These layers typically include an input layer, one or more hidden layers, and an output layer. Each neuron receives input signals (which are usually numbers) from neurons in the previous layer, processes them by applying a weighted sum followed by a non-linear function called an activation function, and then passes the result to neurons in the next layer. The weights represent the strength or importance of the connections, and the activation function introduces non-linearity, enabling the network to model complex relationships rather than just linear ones. Training a neural network involves adjusting these weights so that the network's output matches the desired output as closely as possible. This is done through a process called backpropagation combined with an optimization algorithm like gradient descent. Backpropagation works by calculating the error between the predicted output and the actual output, then propagating this error backward through the network to update the weights in a way that reduces the error over time. Neural networks are particularly powerful because they can automatically learn representations from raw data without needing explicit feature engineering. For example, in image recognition, early layers might detect simple features like edges, while deeper layers identify more complex shapes or objects. There are many types of neural networks designed for specific tasks. For instance, feedforward neural networks process data in one direction from input to output, while recurrent neural networks (RNNs) have loops allowing them to process sequences and remember past information, which is useful in language processing. Another type, convolutional neural networks (CNNs), uses specialized layers to efficiently analyze visual data. Overall, neural networks form the backbone of many modern AI applications, including speech recognition, natural language processing, autonomous driving, and more, because of their ability to learn complex patterns and generalize well to new, unseen data.

2.1.1 Perceptron

THE perceptron is a binary classifier. perceptron is one of the simplest types of artificial neural networks and actually a foundational building block for more complex neural networks. It was introduced in the 1950s by Frank Rosenblatt.

A perceptron takes several input values, each multiplied by a weight that represents the importance of that input. Then, it sums all these weighted inputs and passes the result through an activation function, typically a step function that outputs either 0 or 1. This output represents the perceptron's decision—like classifying data into one of two categories.

Mathematically, if you think of inputs as a vector xx and weights as ww, the perceptron computes:

$$y = \text{activation}\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

where bb is a bias term that shifts the decision boundary.

The perceptron learns by adjusting the weights based on errors between its predicted output and the actual target during training, using a simple update rule. While a single perceptron can only solve

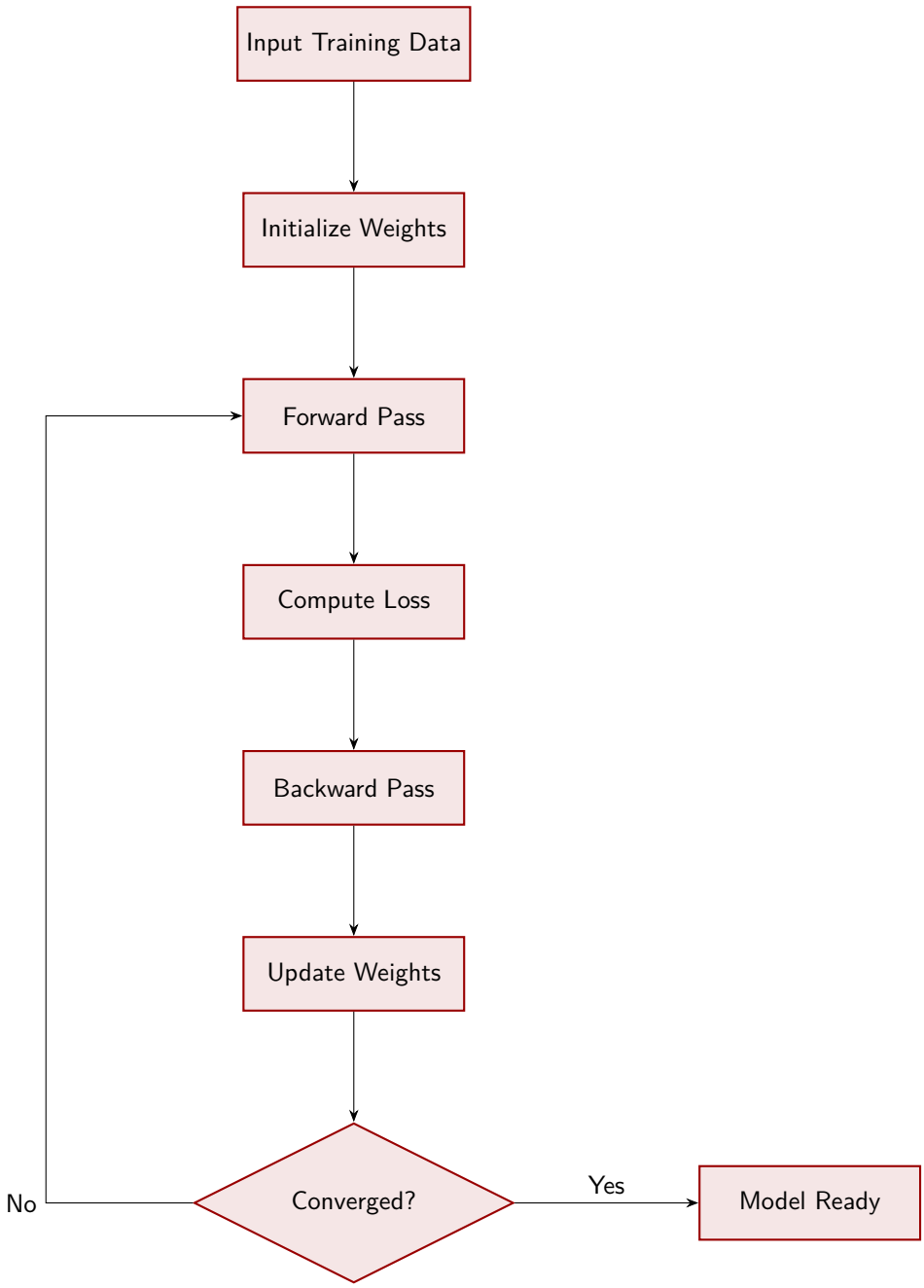
linearly separable problems (where a straight line can separate classes), stacking perceptrons and adding layers leads to more powerful models like multilayer neural networks that can learn complex patterns.

In short, the perceptron is the building block for neural networks, helping machines make simple yes/no decisions and forming the basis for deep learning architectures.

2.2 Training Deep Networks

2.2.1 Backpropagation

Backpropagation is an essential technique. fundamental algorithm in AI and deep learning that enables neural networks to learn from data by adjusting their internal parameters—called weights—in order to improve performance on a given task. When a neural network processes an input, it produces an output, which is then compared to the expected or correct result using a loss function—a measure of how far off the prediction is. The goal of training is to minimize this loss. Backpropagation works by computing the gradient, or the rate of change, of the loss function with respect to each weight in the network. This gradient tells us how much a small change in a particular weight will affect the overall error. The algorithm uses the chain rule from calculus to efficiently calculate these gradients layer by layer, starting from the output layer and moving backward toward the input layer—hence the name "backpropagation." Once the gradients are computed, they guide how the weights should be adjusted. Using an optimization method like gradient descent, the network updates its weights in the direction that most reduces the loss. This process repeats over many iterations (epochs), gradually improving the network's accuracy in making predictions. Without backpropagation, training deep neural networks would be extremely difficult because it allows the model to assign credit or blame to each connection based on its contribution to the final error, enabling the network to learn complex, multi-layered representations from data. This makes backpropagation a cornerstone technique that drives much of the progress in deep learning and modern AI applications.



A more detailed training loop in a deep learning pipeline.

Listing 2.1: A simple neural net in PyTorch

```
import torch
import torch.nn as nn

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc = nn.Linear(10, 1)

    def forward(self, x):
        return self.fc(x)
```

CHAPTER

3

ADVANCED TOPICS

"The true logic of this world is in the calculus of probabilities."

— ***James C. Maxwell***

3.1 GANs and VAEs

GENERATIVE models like GANs are powerful. Generative Adversarial Networks (GANs) are a class of neural networks designed to generate new, realistic data samples that resemble a given dataset. A GAN consists of two neural networks working in opposition—a generator and a discriminator. The generator creates fake data (like images, text, or audio) from random noise, trying to mimic the real data distribution. The discriminator's job is to distinguish between real data (from the training set) and fake data produced by the generator. During training, these two networks play a kind of game: the generator tries to fool the discriminator by producing increasingly convincing fake samples, while the discriminator gets better at spotting fakes. Through this adversarial process, both networks improve over time. The end result is a generator that can produce highly realistic data. GANs have been widely used for tasks like image synthesis, style transfer, and even creating deepfake videos. while Variational Autoencoders (VAEs), on the other hand, are a different kind of generative model that learns to encode input data into a compressed, continuous latent space and then decode it back to reconstruct the original input. Unlike traditional autoencoders, VAEs learn a probability distribution over the latent space rather than fixed points. This allows VAEs to generate new data by sampling from the latent space and decoding it. The key idea behind VAEs is to balance two objectives during training: minimizing the reconstruction error (how close the decoded output is to the original input) and ensuring the latent space follows a known distribution (usually a Gaussian). This makes the latent space smooth and continuous, which helps generate coherent new samples by interpolation. VAEs are often used in tasks such as image generation, anomaly detection, and representation learning. In summary, both GANs and VAEs are powerful generative models but work quite differently: GANs use a competitive training setup between two networks to produce realistic samples, while VAEs use probabilistic encoding and decoding to learn a meaningful latent representation that can be sampled for new data. Each has its own strengths and applications depending on the problem you want to solve.

CHAPTER

4

INTERVIEW QUESTIONS

"The true logic of this world is in the calculus of probabilities."

— ***James C. Maxwell***

4.1 Theory

THIS section covers theoretical concept. deep learning models combines ideas from mathematics and computer science: neural networks approximate complex functions by layering simple mathematical operations, and backpropagation uses calculus to efficiently adjust the network's parameters to reduce errors. GANs rely on game theory, where two networks compete to generate realistic data by learning the true data distribution, while VAEs use principles from probability and Bayesian inference to learn compact, meaningful representations of data and generate new samples by balancing reconstruction accuracy with keeping the underlying data structure organized. Together, these theories enable machines to learn from data, recognize patterns, and create new content in powerful and flexible ways.

4.2 Coding

INTERVIEWERS often ask you to write code. t's their goal is to assess more than just whether you can produce a working program. They want to see your problem-solving process, logical thinking, and how you approach complex challenges step-by-step—which is essentially a form of deep thinking applied in a practical context. Writing code live reveals how you break down problems, choose the right algorithms or data structures, and handle edge cases or errors. It also shows your ability to communicate your reasoning clearly and adapt when faced with unfamiliar scenarios. So, the coding task isn't just about getting the right answer; it's about demonstrating your mindset, technical skills, and capacity to think deeply and logically under pressure, which are critical traits for real-world software development.