



ULTIMATE

Azure AI Services for Gen AI Solutions

Build Advanced Gen AI Solutions
with Azure OpenAI, LangChain and
Vector Databases to Enhance
Efficiency, and Revolutionize
Enterprise Operations

Shanthababu Pandian

Ultimate Azure AI Services for Gen AI Solutions

*Build Advanced Gen AI Solutions with Azure
OpenAI, LangChain and Vector Databases to
Enhance Efficiency, and Revolutionize
Enterprise Operations*

Shanthababu Pandian



www.orangeava.com

Copyright © 2025 Orange Education Pvt Ltd, AVA®

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author nor **Orange Education Pvt Ltd** or its dealers and distributors, will be held liable for any damages caused or alleged to have been caused directly or indirectly by this book.

Orange Education Pvt Ltd has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capital. However, **Orange Education Pvt Ltd** cannot guarantee the accuracy of this information. The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

First Published: May 2025

Published by: Orange Education Pvt Ltd, AVA®

Address: 9, Daryaganj, Delhi, 110002, India

275 New North Road Islington Suite 1314 London,
N1 7AA, United Kingdom

ISBN (PBK): 978-93-48107-46-6

ISBN (E-BOOK): 978-93-48107-31-2

Scan the QR code to explore our entire catalogue



www.orangeava.com

Table of Contents

1. Introduction to Generative AI

Introduction

Structure

Introduction to Generative AI

Evolution of Gen AI

Gen AI - Transformer Architecture

Components of Transformer Architecture

ENCODER

DECODER

Gen AI versus Classical AIML

Machine Learning

Generative AI

Gen AI's Drawbacks

Foundation Models

Large Language Model (LLM)

How Industries Get Benefits out of Gen AI

Gen AI Ethics and High-Level Security

Ethical Considerations Factors

Conclusion

2. Exploring LLMs and Its Capabilities

Introduction

Structure

Overview of Large Language Models (LLM)

Large Language Models (LLMs)

Exploring Available LLMs in the Market

Types of Large Language Models (LLMs)

Applications of Large Language Model (LLM)

Working Principles of Large Language Models

Controlled Output of LLMs and their Significance

Fine-Tuning Process of LLM and their Types

Types of Fine-Tuning

Terminology Closer to LLM During Training

[*When to Opt for Fine-Tuning*](#)
[*Fine-Tuning Benefits*](#)
[Advantages of Large Language Models](#)
[Challenges in Large Language Models \(LLMs\)](#)
[Conclusion](#)

3. Vector Database and Embedding Techniques

[Introduction](#)
[Structure](#)
[Defining a Vector Database](#)
[*Need for a Vector Database*](#)
[*Characteristics of Vector Databases*](#)
[*Vector Database versus Traditional Database*](#)
[*Comparison of Vector and Traditional Databases*](#)
[*Embeddings and Techniques*](#)
[Working Mechanism of a Vector Database](#)
[*Serverless Vector Databases*](#)
[*Vector Embedding and DB-Specific Use Cases*](#)
[Semantic and Similarity Search](#)
[*Types of Similarity Measures*](#)
[Analogy to Understand Vector Database](#)
[*Classification of Vector Databases*](#)
[*Popular Vector Databases in the Market*](#)
[*Working with Pinecone Vector Database*](#)
[*Working with the FAISS Vector Database*](#)
[Conclusion](#)

4. Prompt Engineering and Its Significance

[Introduction](#)
[Structure](#)
[Prompt Engineering and Significance in Gen AI](#)
[*Significance of Prompt Engineering*](#)
[Components and Elements of Prompt](#)
[Role Prompting](#)
[*Role Prompt's Flip Side*](#)
[*Prompting: Voice Specific*](#)
[Prompt Engineering Techniques](#)

[*Prompt Architecture*](#)
[*Best Practices for Building Effective Prompts*](#)
[Conclusion](#)

5. Azure Storage for Azure OpenAI Implementations

[Introduction](#)

[Structure](#)

[Introduction to Azure and Its Environment](#)

[*Understanding Microsoft Azure*](#)

[*Azure Resource Management*](#)

[*Best Practices for Azure Storage*](#)

[*Creating Azure Resource Group*](#)

[*Creating an Azure Storage Account for Blob*](#)

[*Create a Blob Container*](#)

[*Creating a Storage Account and ADLS Gen2*](#)

[Conclusion](#)

6. Azure AI Search Services for Azure OpenAI Implementations

[Introduction](#)

[Structure](#)

[Overview of Azure AI Search Services](#)

[Capabilities of Azure AI Search Services](#)

[Azure AI Search Services: An AI Requirement](#)

[*Exploring Azure AI Search Services*](#)

[*Azure Storage and AI Search Services Integration*](#)

[*Creating an Azure Storage Account for Blob*](#)

[*Creating Azure AI Services*](#)

[Conclusion](#)

7. Getting Started with Generative AI Using Azure OpenAI Services

[Introduction](#)

[Structure](#)

[Understanding OpenAI](#)

[*Essential Concepts in OpenAI*](#)

[Significance of AOAI Services in Gen AI](#)

[*Enhanced Innovation Tools*](#)

[*Microsoft AOAI Capabilities*](#)

[*Exploring AOAI*](#)

[*Understanding the AOAI Architecture Workflow*](#)

[*AOAI Resource Creation*](#)

[Concepts in Azure Open AI Studio](#)

[*Creating an Azure Storage Account for Blob*](#)

[*Creating Azure AI Services*](#)

[Azure OpenAI Studio: Add your Data and Configuration](#)

[Deployment: Chatbot](#)

[Playground: Completions](#)

[Understanding the Environment](#)

[DALL-E Playground](#)

[Conclusion](#)

8. Advanced Azure AI Studio–I

[Introduction](#)

[Structure](#)

[Azure AI Studio Invents a New Era of Gen AI Development](#)

[*Core Capabilities of Azure AI Studio for Gen AI*](#)

[Azure AI Studio Empowerments](#)

[*Azure AI Studio Architecture*](#)

[Azure AI Studio and its Environment](#)

[*API and Model Choice*](#)

[*Prompt Catalog*](#)

[*Responsible AI Initiatives*](#)

[*Understand Responsible AI using Azure AI Studio*](#)

[Prompt Flow](#)

[*Prompt Flow Architecture*](#)

[*Building Prompt Workflow*](#)

[Azure AI Content Safety](#)

[Conclusion](#)

9. Advanced Azure AI Studio–II

[Introduction](#)

[Structure](#)

[Deep Dive into Azure Speech Studio](#)

[*Azure Speech SDK*](#)

[*Speech-to-Text Implementation Using Azure Speech SDK Python*](#)

Features of Azure AI Speech Studio (Text-to-Speech)

Speech-To-Text

Significance of Azure AI Speech

Real-Time Applications with Azure AI Speech

Speech Synthesis Markup Language (SSML)

Deep Dive into Azure Vision Studio

Azure Vision Studio: Features

Spatial Analysis

Azure Image Analysis SDK

Image Analysis: Implementation using Azure Vision SDK Python

Deep Dive into Azure Language Studio

Key Features of Azure Language Studio

Microsoft Visual Studio Code

Best Practices for VS Code Environment

Significance of VS Code as Azure AI SDK

Comparative Study of VS Code with Other SDKs

Azure AI Foundry

Azure AI Foundry Architecture

Azure AI Foundry Project

Conclusion

10. Generative AI Use Cases for Industries–I

Introduction

Structure

Necessities of Gen AI in Various Industries

Gen AI in the Banking and Financial Industry

Customer-focused Gen AI Scope and Opportunity

Bankers-focused Gen AI Scope and Opportunity

Gen AI in the Healthcare Industry

Gen AI in the Entertainment and Media Industry

Gen AI in the Manufacturing and Industry 4.0

Gen AI in the Agriculture and Food Industry

Ethical Consideration in Gen AI

Ethical Consideration in Banking and Financial Industry

Ethical Consideration in the Healthcare Industry

Ethical Consideration in the Entertainment and Media Industry

Ethical Consideration in Manufacturing and Industry 4.0

Ethical Consideration in the Agriculture and Food Industry
Discussion on Security Aspects of Gen AI
Conclusion

11. Gen AI Implementation Use Case with Azure OpenAI–II

Introduction

Structure

Azure OpenAI Solutions for Business Advantage

Industry-Centric Benefits of AOAI Solutions

Healthcare Use Cases Using Azure OpenAI

Architectural Design and Development for (Med-Docs and Summary) Using Azure Ecosystem

Python Code for Med-Docs and Summary Implementation

Manufacturing Use Cases Using Azure OpenAI

Architectural Design and Development for (Predictive Maintenance) Using Azure Ecosystem

Python Code for Predictive Maintenance Implementation

Banking and Financial Services Use Cases Using Azure OpenAI

Architectural Design and Development for (Fraud Detection and Prevention) Using Azure Ecosystem

Python Code for Fraud Detection and Prevention Implementation

Education Domain Use Cases Using Azure OpenAI

Architectural Design and Development: Knowledge Base and Information Retrieval Using Azure Ecosystem

Python Code for Knowledge Base and Information Retrieval Implementation

Conclusion

Index

CHAPTER 1

Introduction to Generative AI

Introduction

This chapter will discuss Generative AI and the evolution of Artificial Intelligence (AI) as well as transformer architecture backgrounds. It will then discuss how Gen AI differs from classical AI and ML. The core of Generative AI and its characteristics transformed text, images, and video processing using Gen AI. We should remember that not all Gen AI models are Large Language Models (LLMs), but all LLMs are a type of Gen AI. Moreover, LLMs specifically refine how we generate and interact with the given textual data. The theories and rules behind these models and how they relate to neural networks, training approaches and text generation are equivalent to human-like content. Why do industries require Gen AI, and what are the significant benefits along with Gen AI ethics and high-level security aspects of this technology?

Structure

In this chapter, we will discuss the following topics:

- Introduction to Generative AI
- Evolution of Gen AI
- Generative AI and its Characteristics
- Gen AI Transformer Architecture
- How Gen AI Differs from Classical AI and ML
- How Industries Get Benefits out of Generative AI
- Gen AI Ethics and High-Level Security

Introduction to Generative AI

In recent years, Generative AI has become the buzzword in this busy world. Every industry and every commoner talks about AI in all forms. In this book, Generative AI, or, shortly, Gen AI, will be discussed at a very detailed level, and experiments

will be conducted to build the Gen AI application using various tools and technologies.

Definition of Gen AI – A simple one-line answer is as follows:

“The application will generate new information for you from your input.”

We cannot treat all this information as simple and take it lightly. It is compelling and insightful and provides solutions for various problem statements and decision-making purposes for numerous industries’ specific challenges in multiple aspects. This is the main reason all industries are focusing on building their own Gen AI tools and integrating multiple AI tools to accomplish their needs and necessities.

It does not matter what input format you use; simultaneously, you can generate your desired output format. We will discuss the input and output type and format in detail in the upcoming chapters and how technologies are helping to achieve this. [Figure 1.1](#) shows the simple block diagram of the Gen AI System.

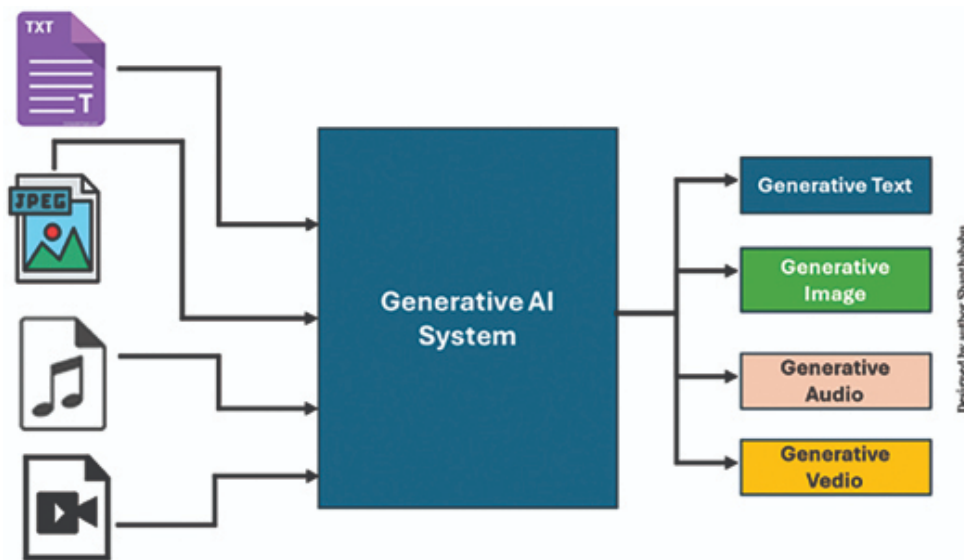


Figure 1.1: Simple Block Diagram of Gen AI

Technically, we can define Gen AI as a branch of “*Artificial Intelligence*” that involves using specific algorithms and larger models to create new and original content from input. This category can generate content that is not replicated or copied from the original input data and shares the relevant characteristics. Undoubtedly, the end product is entirely new and relevant to business aspects.

Characteristics of Gen AI Involving LLMs:

The characteristics of Gen AI involving LLMs are as follows:

- Gen AI has non-deterministic outputs and creates new and original content based on the business point of view.
- This involves a complex training mechanism to generate synthetic data for various business perspectives.
- It consists of the massive number of Neural Networks involved internally to generate accurate data for analysis.
- It adds more creative elements to generate precise content to attain the expectations.
- Ultimately, it helps to generate additional training data to support unbalanced situations.

Benefits of LLMs:

- We can build very creative and innovative products for business solutions.
- We can enhance the customer experience journey from past collections of data to strengthen customer bonding.
- We can construct high-time and cost-effective products from vast quantities of data.
- We can customize the products to be highly scalable to achieve the results.
- We can ease the process of data analysis with less effort by using tools and technologies.
- We can also automate the task behind the screen with practical solutions, and reduce manual interventions.

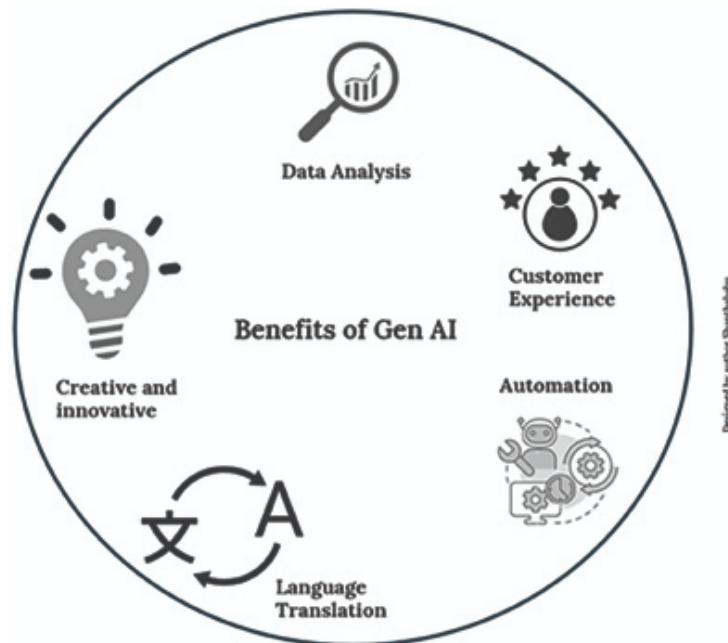


Figure 1.2: Benefits of Gen AI

Evolution of Gen AI

The popularity of Gen AI has been around for decades, but its actual breakthrough in the industry is subsequent to the introduction of ChatGPT in the market. The main objective is to generate the content, whether text, image, audio, or video. Gen AI is an exceptionally revolutionary technology that brings abundant benefits to businesses irrespective of any domain.

The Evolution of Gen AI started in the 1950s and has had a long journey with significant milestones.

1952—Love Letter: British computer scientist Christopher Strachey built a software program to generate Love Letter text. It was considered one of the earliest examples of creativity and natural language generated by TEXT.

1965-66—First Chatbot: During 1965-66, Joseph Weizenbaum, a computer scientist at the MIT Artificial Intelligence Laboratory, built the first chatbot application. It was called ELIZA. It was designed to simulate a discussion with a Rogerian psychotherapist using the concepts behind pattern-matching approaches and scripted-based responses.

This chatbot is a pioneer for pre-programmed responses based on those inputs. It paves the way for natural language processing, human-computer interaction, future chatbots, and conversational agent developments.

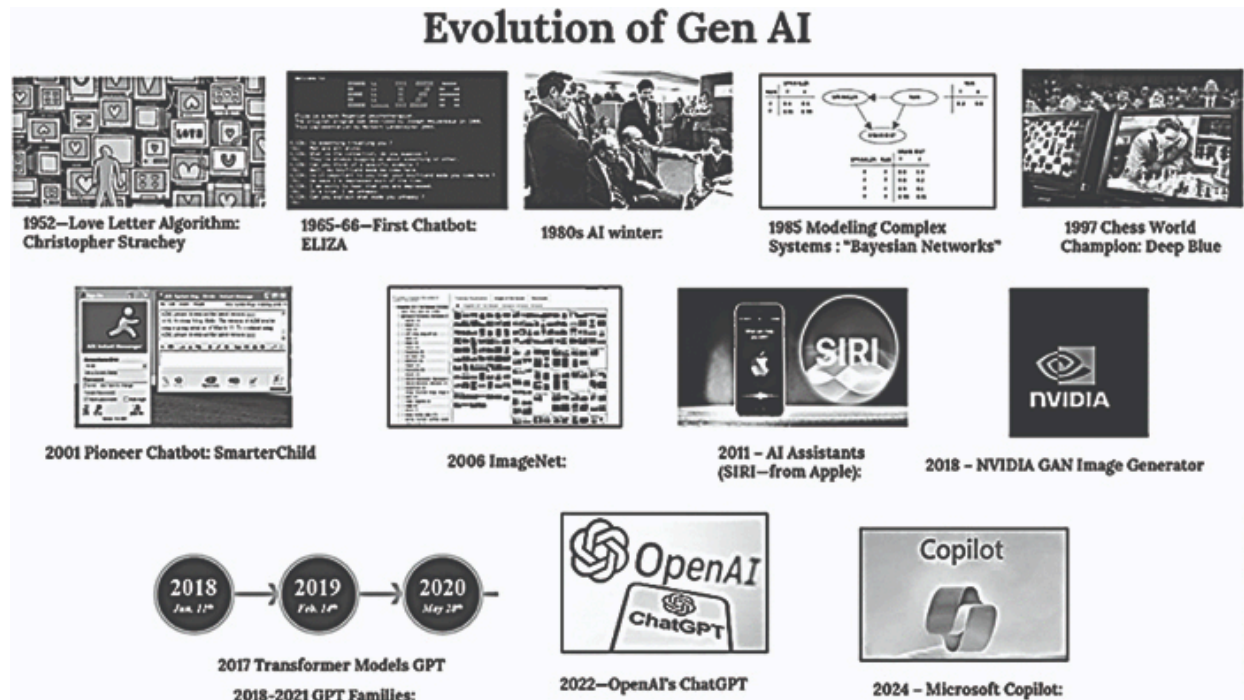


Figure 1.3: Evolution of Gen AI

1980s AI Winter: In AI history, there was a slowdown in this technology during the 1980s. During this period, action was taken to reduce funding and interest in AI research and development, and typically, in other fields, hype, disappointment, and critique occurred in the AI domain, resulting in funding cuts. However, work continued in the area of neural networks.

1985 Modeling Complex Systems: During this period, Judea Pearl, a computational scientist and a great philosopher, introduced typical mathematical-based “*Bayesian Networks*” that seeded for the complex system modeling in the Generative AI era.

1997 Chess World Champion: The famous IBM chess computer “Deep Blue” defeated the chess world champion Garry Kasparov. This is the best evidence of AI’s decision-making strategy and the event that unlocks the potential of Generative AI.

2001 Pioneer Chatbot: In 2001, Active Buddy developed the “**SmarterChild**” chatbot version of the Generative AI application and launched it on AOL’s (American Online) Instant Messaging (AIM) platform. One of the pioneering CHATBOTS extensively used by most consumers, the product was designed to provide information regarding entertainment and services to major AIM users.

2006 ImageNet: Dr. Fei-Fei Li at Stanford University invented the “ImageNet” product, which utilizes a larger volume of visual databases to train the computer vision algorithm for object detection and recognition, which involves a more significant contribution of Deep Learning and AI.

2011 – AI Assistants (SIRI—from Apple): In 2011, Apple introduced the AI Assistants (SIRI). It revolutionized voice-based assistance and was the first product that came into public hands as a highly convenient and practical device for houses. It supports various aspects of assistance, such as Boss meeting reminders, voice recognition communication, and integrating and controlling other devices based on the command.

2017 Transformer Models: Open AI organization released a Generative Pre-Trained Transformer (GPT-1) language model that helps to generate the text based on the given input prompt.

2018 – NVIDIA GAN Image Generator: The breakthrough of NVIDIA’s AI image generator is a landmark in the AI art creation environment. It introduces the StyleGAN architecture, which can generate highly creative, realistic, customizable images based on needs. Due of its compact size, training time, and excellent operation, all these abilities are set to transform the way we use Generative AI.

2018-2021 GPT Families: During this period, the extension of transmission models (GTP-3 and 4) expanded further along with the massive parameters inputs and generated creative human-like text and images at different levels and incomparable height of the technology expansion in AI space.

2022—OpenAI’s ChatGPT: After the introduction of ChatGPT, Gen AI utilization increased enormously since its capabilities include unlivable content creation, precise answers, and more interaction with humans. Nowadays, everyone has an account in ChatGPT.

2024 – Microsoft Copilot: As we know, Microsoft is always unique in technology adoption, no exception in this Gen AI; they introduced the Copilot by integrating ChatGPT with Microsoft 365, which is a significant milestone and brings excellent features for many business use cases. This is a major release from Microsoft’s Azure OpenAI Studio with unique features to build highly intelligent tools and technologies for many industries’ innovative solutions. Indeed, we are going to cover most of them in this book.

Note:

Evolution of Gen AI

1952—Love Letter: Christopher Strachey, a British Computer Scientist.

1965-66—First Chatbot: During 1965-66, Joseph Weizenbaum, a computer scientist, built the first chatbot application.

1980s- AI winter: In AI history, there was a slowdown in AI technology.

1985- Modeling Complex Systems: During this period, Judea Pearl, a computational scientist, and great philosopher, introduced typical mathematical-based systems.

1997- Chess World Champion: The famous IBM chess computer “Deep Blue” defeated the chess world champion Garry Kasparov.

2001- Pioneer Chatbot: In 2001, Active Buddy developed the “SmarterChild” chatbot version on AOL’s (American Online) Instant Messaging (AIM) platform.

2006- ImageNet: Dr. Fei-Fei Li at Stanford University invented the “ImageNet” product, a computer vision algorithm for object detection and recognition.

2011– AI Assistants (SIRI—from Apple): In 2011, Apple introduced the AI Assistants (SIRI).

2017-22- Transformer Models: GPT Families and OpenAI’s ChatGPT.

2024– Microsoft Copilot: As we know, Microsoft is always unique in technology adoption.

Gen AI - Transformer Architecture

Transformer Architecture Overview:

One of the Generative AI models is the **Transformer-Based Model**, which primarily adopts neural network techniques and natural language processing methodologies, such as text generation, sentiment analysis, question-and-answer, text summarization, and language translation.

In simple terms, the transformer architecture works as a Football team in which the players have the connection and thought process of passing the ball from one side of the court to the target to hit the ball inside the goalpost. This is a very high collaborative effort incurred process.

Since these transformer models are built from the deep neural network-based mathematical model, their structure and function replicate the human brain system, which consists of multiple layers of neurons. So, each neuron can receive various input data, perform a powerful mathematical computation internally with the data, and process the influential output.

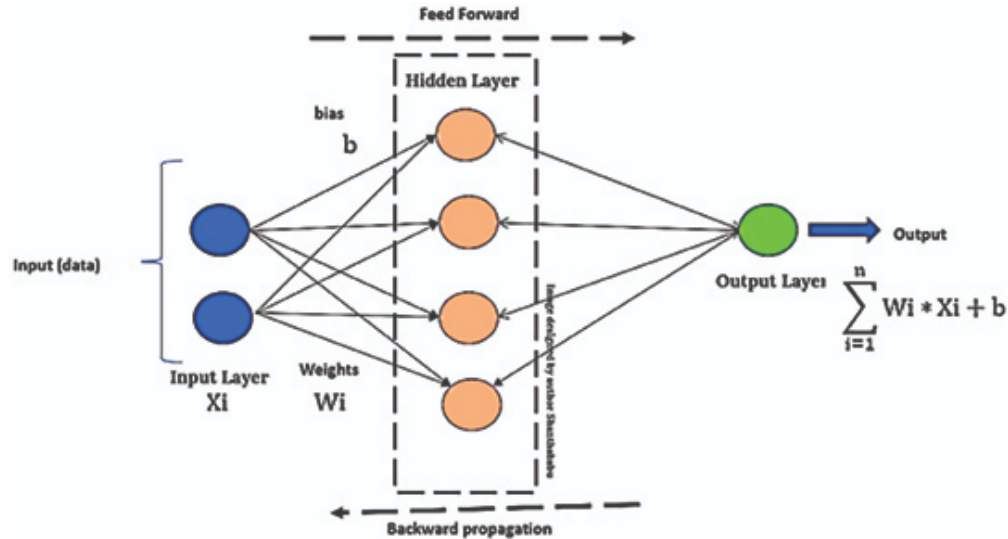


Figure 1.4: Neural Network

The transformer model is highly capable of generating high-quality text in various contexts based on the demand and requirements. This model uses a **SELF-ATTENTION MECHANISM**, a distinctive feature that captures long-range dependencies and is more effective than traditional Deep Learning and NLP models.

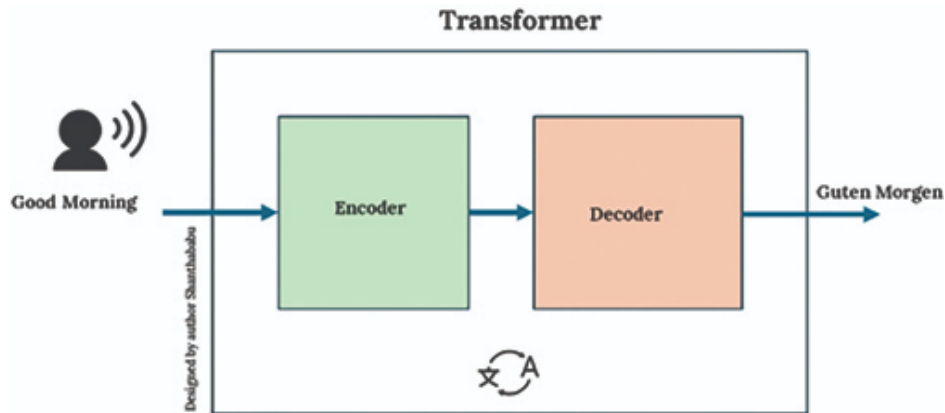


Figure 1.5: Simple Language Translation Transformer Representation

Components of Transformer Architecture

Let us discuss the components of Transformer Architecture:

At a very high level, this architecture uses two major pillars: **Encoder** and **Decoder**.

- **ENCODER:** It takes the input feed, extracts context, and then the meanings from the data, which it passes onto the decoder.
- **DECODER:** It is responsible for converting the input into several sequences from the encoder's feed. These embeddings are processed by the SELF-ATTENTION technique. The decoder's architecture is similar to the encoder and produces the generative output.
- **SELF-ATTENTION:** Self-attention is an important process stage. In this stage, it captures the significance of each word and the relationships between them. This is typically how humans experience understanding what verbs, nouns, and adjectives are. Nouns are when we speak and write and the thought process of how each word relates to every other word in a sentence.

The rest of the components are Input and Output.

- **INPUT:** This is just an input for the system.
- **OUTPUT:** The output is the sequence generated by the transformer.

Let us think big. [Figure 1.6](#) shows the Transformer Architecture (Universal Picture). There is nothing new we cannot add here.

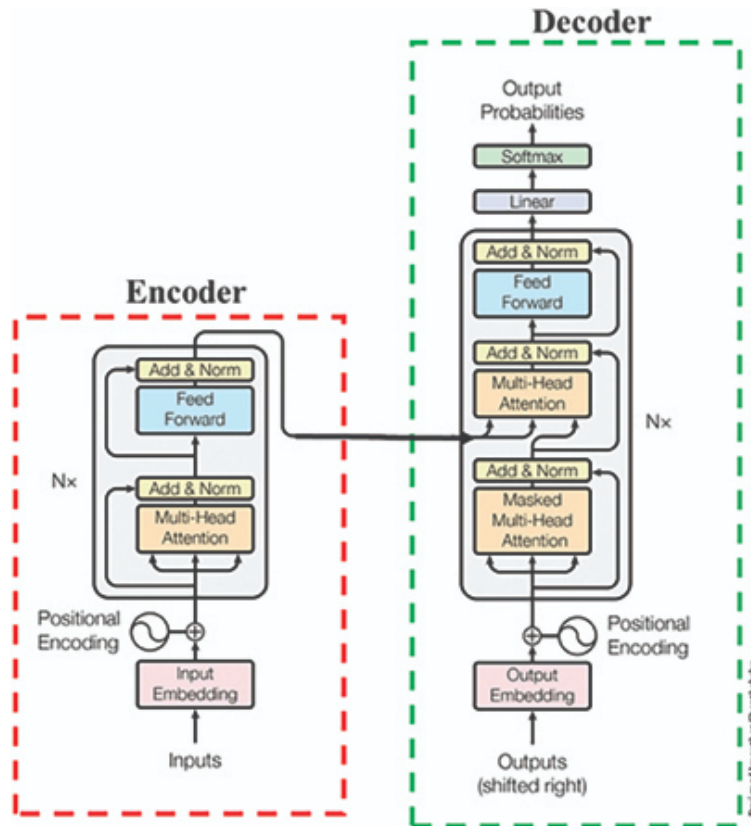


Figure 1.6: Transformer Architecture (Universal Picture)

As we understood from the earlier explanation, the “**Transformer Architecture**” follows an **ENCODER-DECODER** structure and does not depend on the model, such as the Recurrence and Convolutions, to generate an output.

The left part is **ENCODER**, and the right part is **DECODER**.

ENCODER

The Encoder consists of N identical layers. Each layer contains a **MULTI-HEAD ATTENTION LAYER** and a **FEED FORWARD NEURAL NETWORK** as a fully connected set of blocks. The output of each layer is used as an input to the subsequent layer with the same dimensionality. The block count depends on the value of x . If $x=8$, there are $N \times 8$ blocks and layers inside the encoder at the left part of the encoder.

As per the mechanism, the **ENCODER** maps a set of inputs as sequences, and continuous inputs are representations. They undergo the $N \times$ block and layer with x times the “Addition and Normalization” process, which is then fed into a **DECODER**; refer to [Figure 1.6](#).

Before getting into the encoder, let us understand what **Input Embedding** and **Positional Encoding** are.

Even the AI system cannot understand the word(s) directly, so we have to convert them into numbers prior to feeding them into the system; the mechanism of this conversion should be implemented in sensible ways to build an input sequence into the system.

This process is called **Input-Embedding**, the first component in the encoder's block. You will see it in [Figure 1.6](#), in which the input is transformed into sensible numbers, constructed in multidimensional vector space, and fed into Multi-Head Attention (MHA) later than positioning them with the help of the positional encoding process. Let us discuss the following steps:

Step 1: We have to organize the input sentence into a comma-separated list of the words.

Step 2: Add two additional tokens to indicate the start and end of the sentence.

Step 3: Follow the vocabulary list, and have to map each vocabulary with a unique number; this is the so-called indexing.

Step 4: As per the standards, we have to normalize all these words as **One Hot Encoding** to build the high-dimensional vector base. The process of converting each word into these high-dimensional vectors is called **Input Embedding** or **Token Embedding**.

Step 5: The Transformer works differently from LSTM, where the input is fed into the system parallelly. This means the order of the sentence might be lost. Technically, adding additional information of the same dimension as the input vector to represent the sequence of the words in the sentence, which is called **Positional Encoding**.

So, finally, the combination of input embedding and positional encoding vectors will be fed into Multi-Head Attention (MHA).

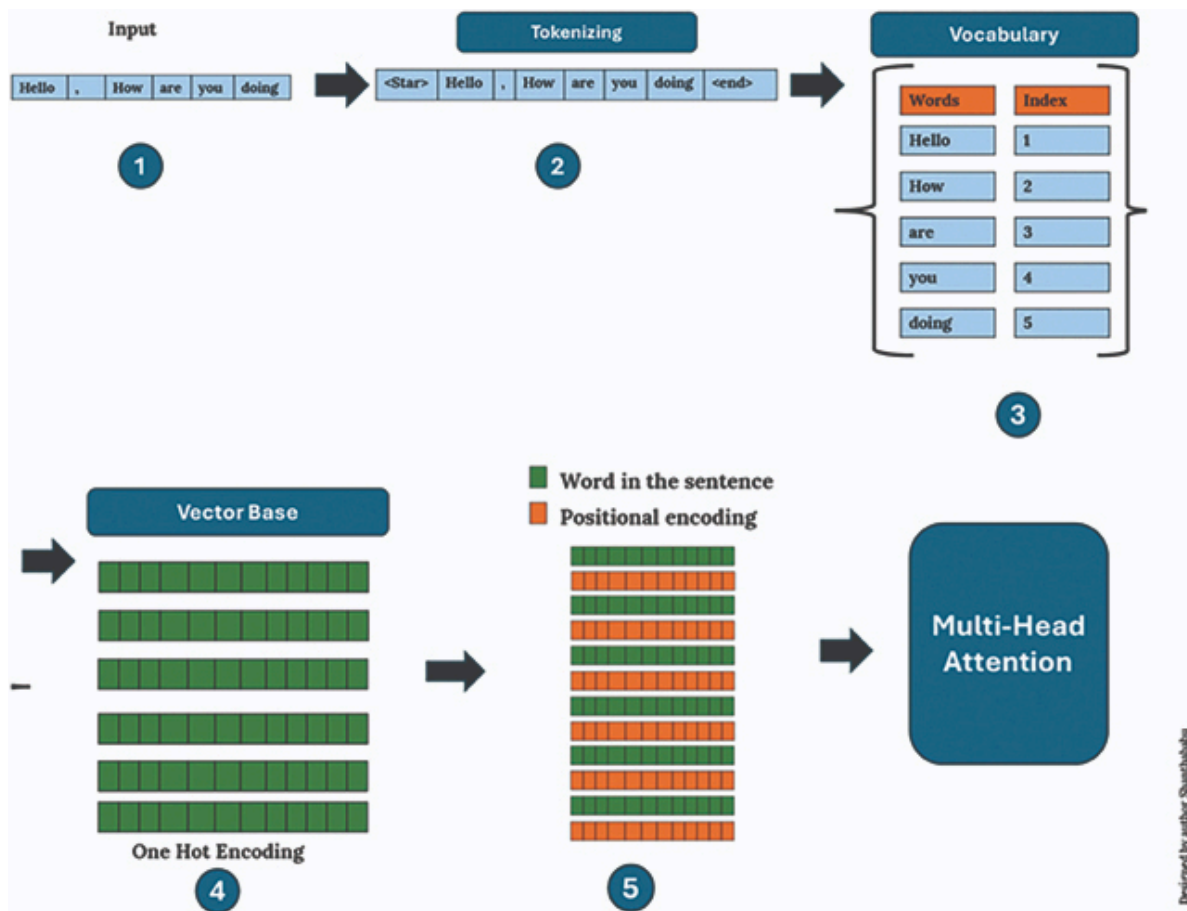


Figure 1.7: Input and Positional Embedding in Transformer Architecture

The Positional Encoding Layer is a Matrix, because: The positional encoding scheme in transformer architecture has each position versus index mapped to a vector. Therefore, the output of the positional encoding layer is a matrix.

Let us do a little deep dive into the **Single and Multi-Head Attention layers**.

As we can see, the attention layers are in the encoder and decoder, which are part of the transformer architecture.

Stage 1: Tokenization and Embeddings: This is the pre-stage in the attention layer building; at this stage, the given input text has been tokenized, and the embedding of each word in the input query, as shown in [Figure 1.8](#).

Stage 2: Vectorization: In this stage of the attention layer, which consists of three neural networks, each word produces three output vectors, such as Queries, Keys, and Values. This process takes place for each word in the given input query.

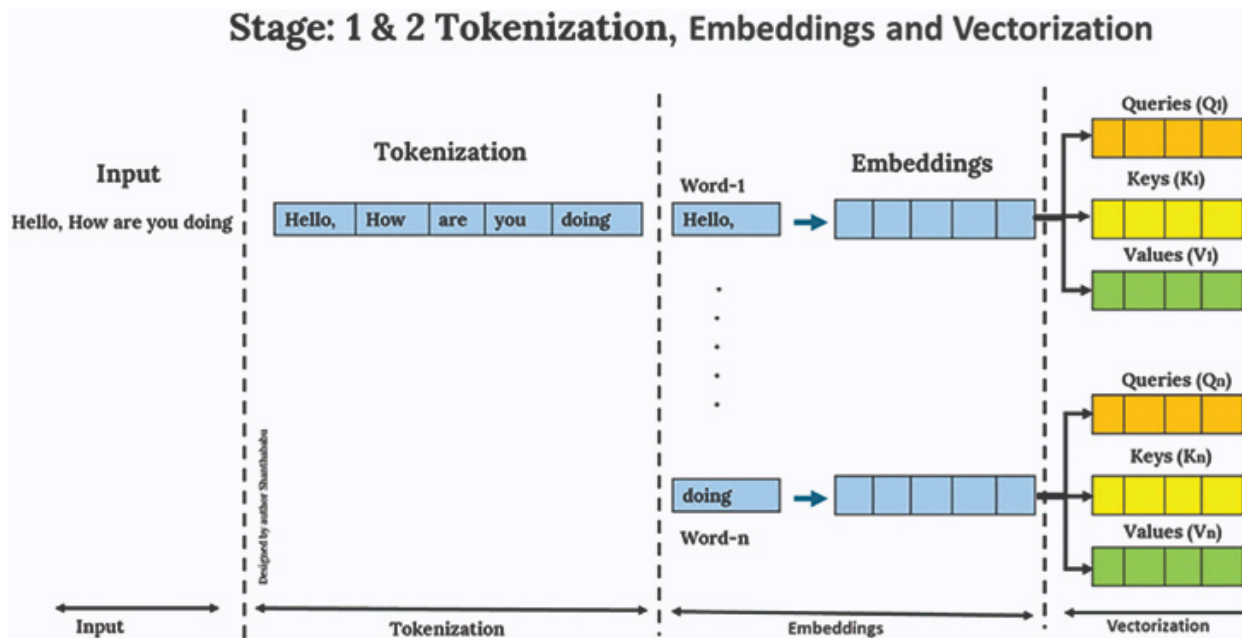


Figure 1.8: Stage 1 and 2 Attention - Tokenization, Embeddings, and Vectorization

Stage 3: Calculating the Word's Score: In this stage, the score of each word is calculated by multiplying Q and K . The process starts with the dot product of the first word's Vector Query (Q) and all the Vectors of K of the input.

This process will confirm the similarity measure of each product of Q and K .

The exercises will start the dot product between the first word of the query (Q_1) and all keys ($K_1 \dots K_n$) in the input sentence, which are used to produce the scores.

If the Q and K vectors are aligned and have meaningful values, their dot product will produce a significant value as an outcome.

If the Q and K vectors are close to orthogonal, then the result of the dot-product will approach zero.

You've Just Finished your Free Sample

Enjoyed the preview?

Buy: <http://www.ebooks2go.com>