Where to get started with GenAI

PRIYANKA VERGADIA AND ALEX XU

JUL 16, 2024

572 12 43 Sha

Introduction to Generative AI

The world of Generative AI (GenAI) is moving at a breakneck pace.

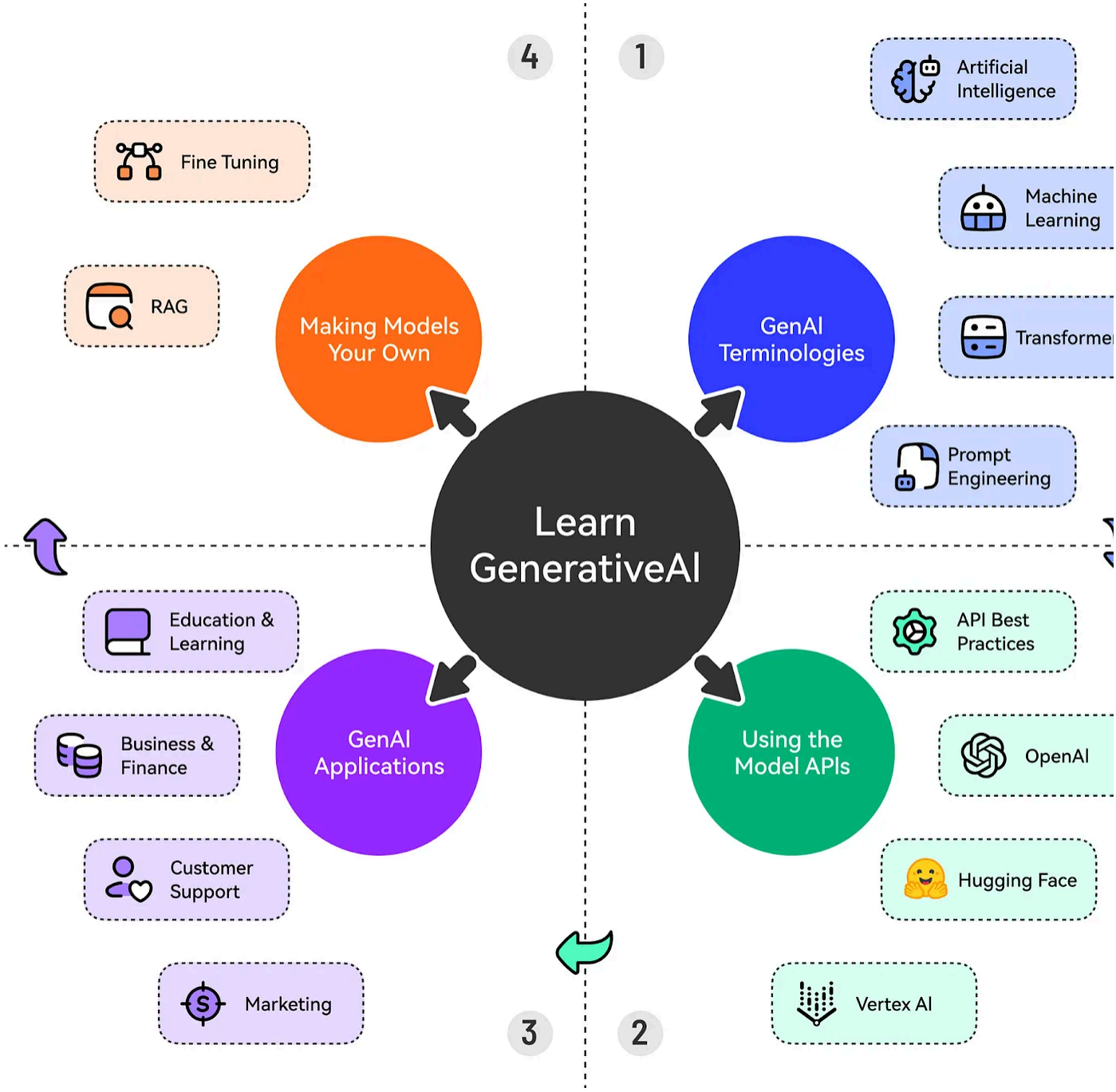
New models, techniques, and applications emerge every day, pushing the boundarie of what's possible with artificial intelligence.

Considering this fast-evolving landscape, developers and technology professionals need to keep their skills sharp and stay ahead of the curve.

To help you get started with GenAI, Priyanka Vergadia and I have put together a concise guide covering essential steps, including:

- Understanding the terminologies
- Using the Model APIs
- Building applications using the AI Models
- Making models your own:
  - RAG
  - Fine-Tuning AI models

Here’s a sneak peek at all the cool topics we will cover.



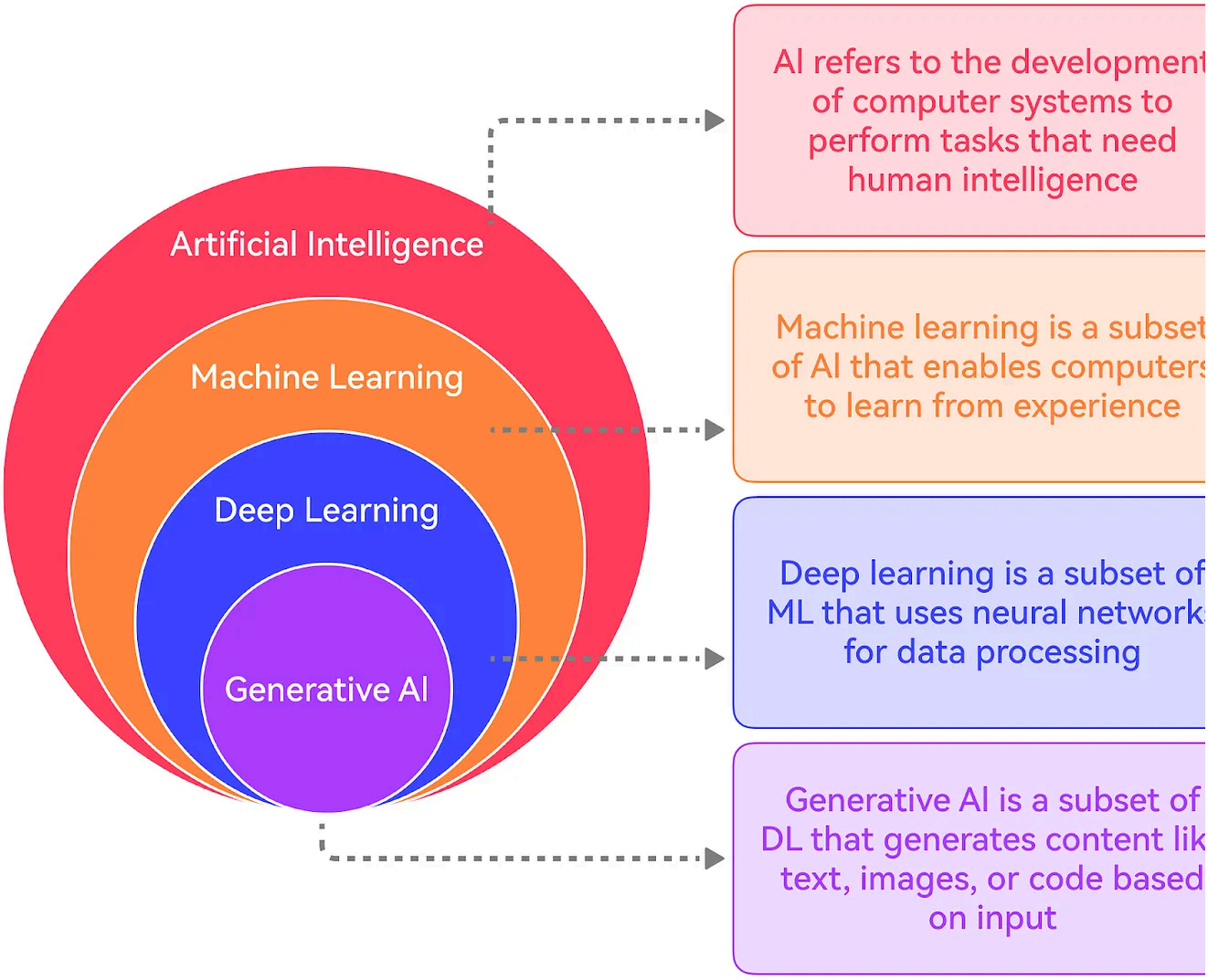
Let's start with the first step.

Also, don't forget to follow [Priyanka Vergadia's LinkedIn](), which is a must-read for anyone working on cloud and GenAI.

Understanding the GenAI Terminologies

One of the biggest obstacles to getting started with GenAI is not understanding the basic terminologies.

Let's cover the most important things to know about.



Artificial Intelligence

AI refers to the development of computer systems that can perform tasks that typically require human intelligence. It is a discipline like Physics.

It encompasses various subfields, such as Machine Learning, Natural Language Processing, Computer Vision, etc.

AI systems can be narrow (focused on specific tasks) or general (able to perform a w range of tasks).

Machine Learning

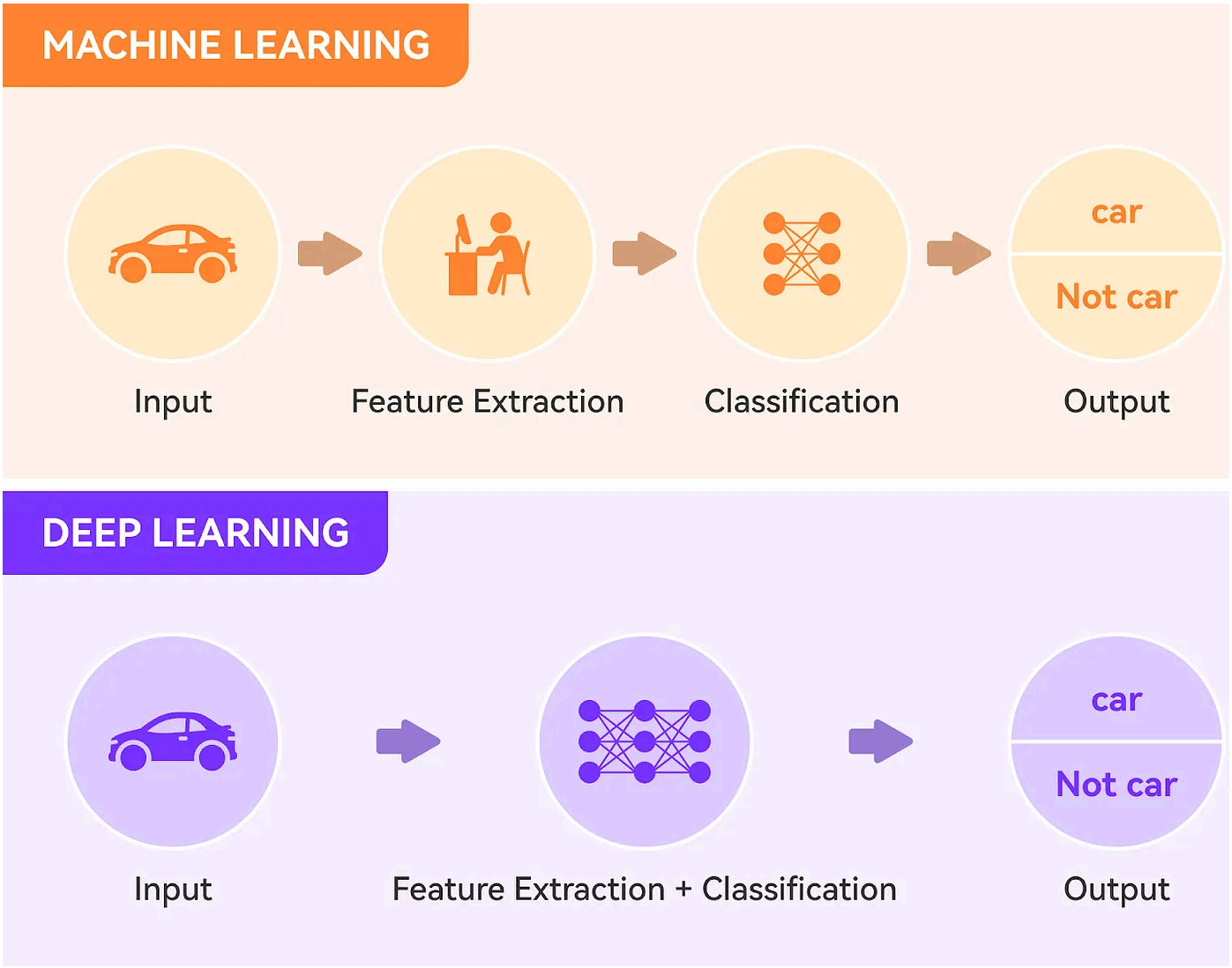
Machine Learning is a subset of AI that focuses on enabling computers to learn and improve from experience without being explicitly programmed.

It involves training models on data to recognize patterns, make predictions, or take actions. There are three main types of ML:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning.

Lastly, there is Deep Learning, which uses artificial neural networks and is a subfie of Machine Learning.

The diagram below shows the key difference between a typical machine learning workflow and Deep Learning.



Natural Language Processing (NLP)

NLP is a subfield of AI that focuses on enabling computers to understand, interpret and generate human language.

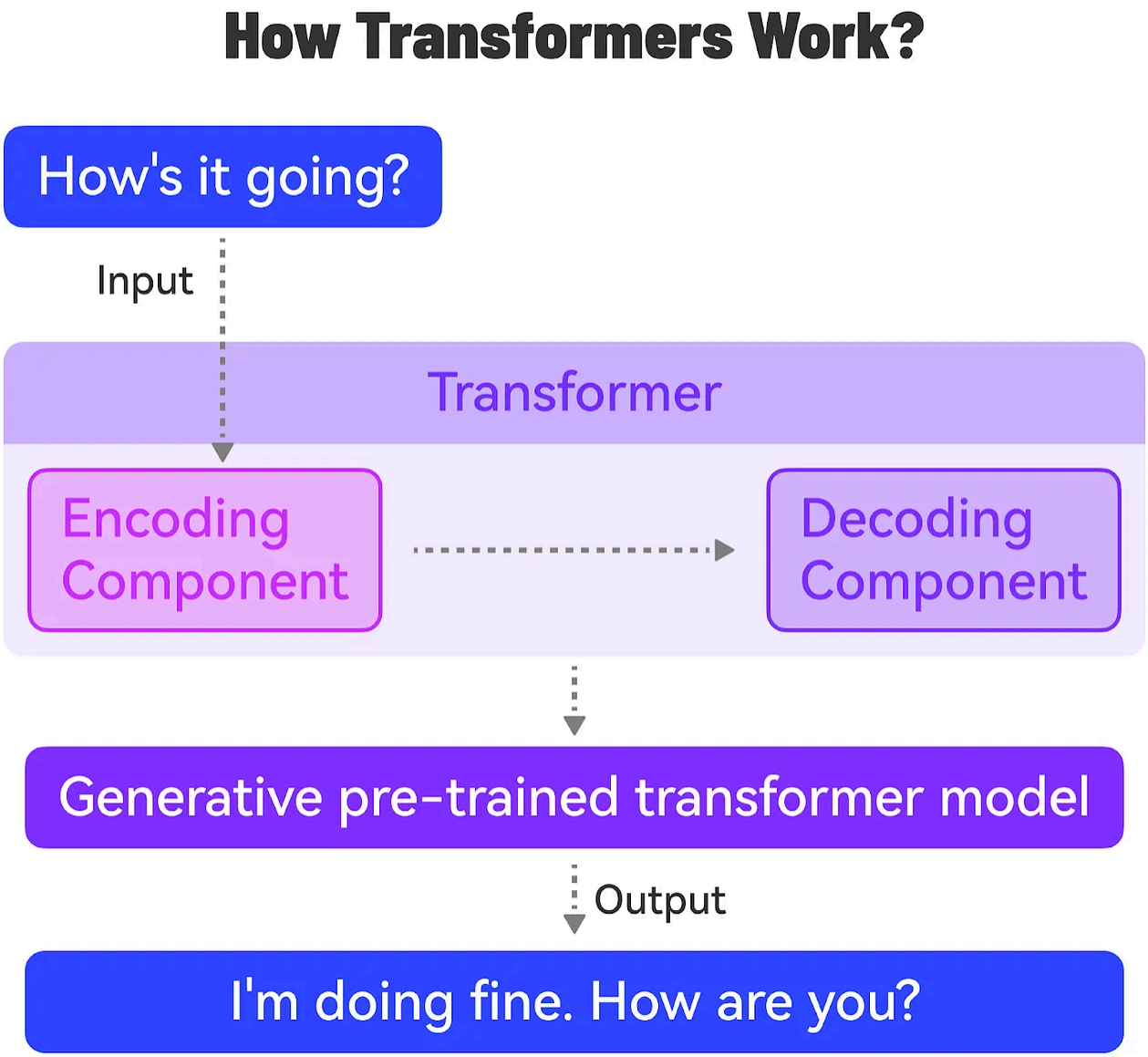
It involves tasks such as text classification, sentiment analysis, entity recognition, machine translation, and text generation.

Deep learning models, particularly Transformer models, have revolutionized NLP in recent years.

Transformer Models

Transformer models are a type of deep learning model architecture introduced in th famous paper “Attention is All You Need” in 2017.

They rely on self-attention mechanisms to process and generate sequential data, su as text.



Transformers have become the foundation for state-of-the-art models in NLP, such BERT, GPT, and T5. They have also been adapted for other domains, like computer vision and audio processing.

GenAI

GenAI, short for Generative Artificial Intelligence, refers to AI systems that can generate new content, such as text, images, or music. It can be considered a subset Deep Learning.

GenAI models can generate novel and coherent outputs that resemble the training data. They use machine learning models, particularly deep learning models, to learn patterns and representations from existing data.

NLP is a key area of focus within GenAI, as it deals with generating and understanding human language. Transformer models have become the backbone of many GenAI systems, particularly language models.

The ability of Transformers to learn rich representations and generate coherent tex has made them well-suited for GenAI applications. For reference, a transformer mo is a type of neural network that excels at understanding the context of sequential da such as text or speech, and generating new data. It uses a mechanism called "attention" to weigh the importance of different parts of the input sequence and be understand the overall context.

There are various types of GenAI Models:

1. Language models that specialize in processing and generating text data. Examp include Google's Gemini, GPT-4, Claude Opus, Llama3
2. Multimodal Models that can handle multiple modalities, like text, images, and audio. Examples include DALL-E, Midjourney, Stable Diffusion.
3. Audio Models that can generate and process speech, music, and other audio da Examples: Google's Imagen, Wavenet.

Prompt Engineering

Prompt engineering is the practice of designing effective prompts to get desired outputs from GenAI models. It involves understanding the model's capabilities, limitations, and biases.

Effective prompts provide clear instructions, relevant examples, and context to guic the model's output.

Prompt engineering is a crucial skill for getting the most out of GenAI models.

Using the Model APIs

Most Generative AI (GenAI) models are accessible through REST APIs, which allov developers to integrate these powerful models seamlessly into their applications.

To get started, you'll need to obtain API access from the desired platform, such as Google's Vertex AI, OpenAI, Anthropic, or Hugging Face.

Each platform has its process for granting API access, typically involving

- Signing up for an account
- Creating an API key
- Completing a verification or approval process.

Once you have your API key, you can authenticate your requests to the GenAI mod endpoints.

Authentication usually involves providing the API key in the request headers or as parameter. It's crucial to keep your API key secure and avoid sharing it publicly.

It's also important to follow best practices to ensure reliability and efficiency. Here a couple of important best practices:

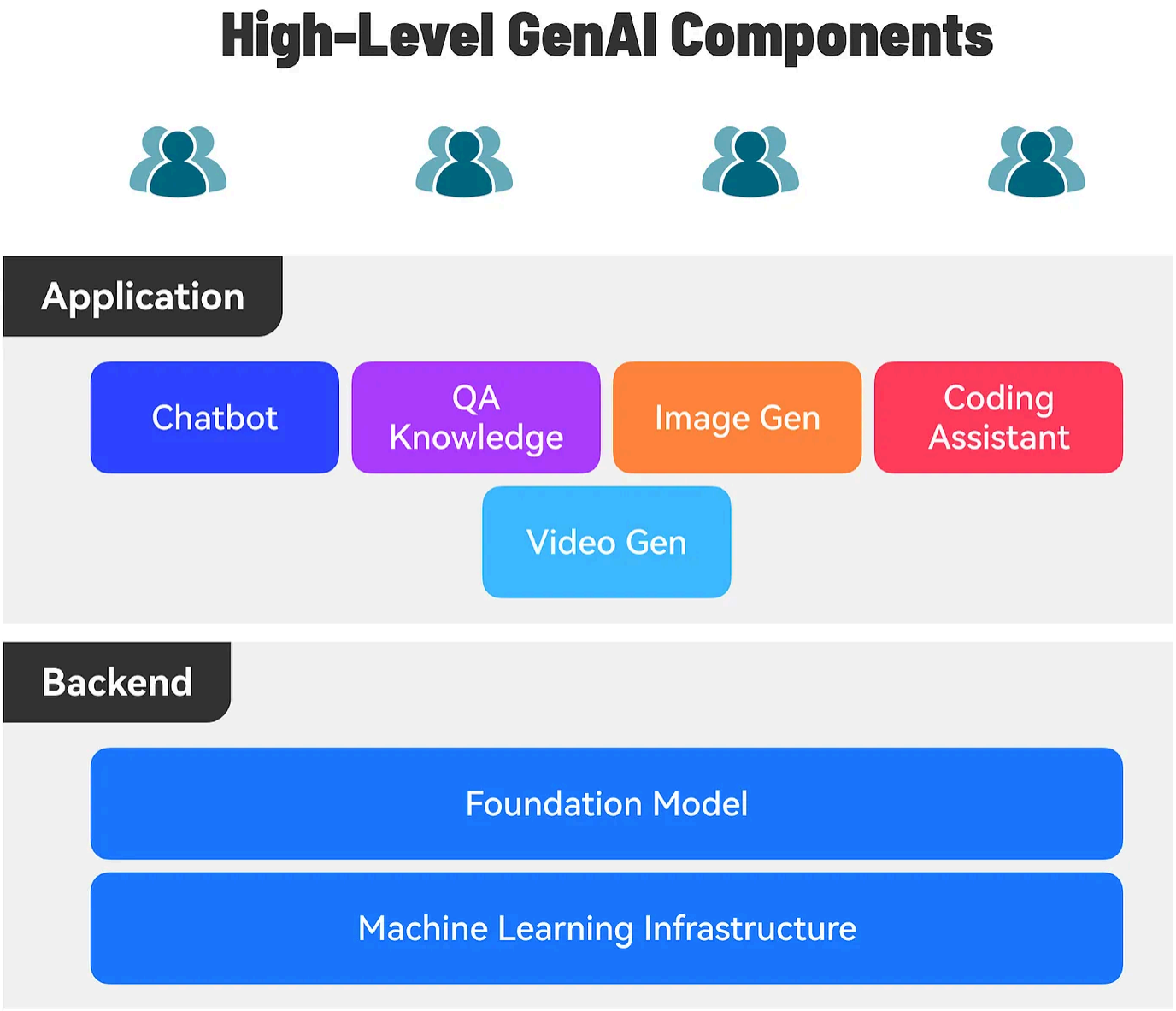
- Handle API errors gracefully by checking the response status code.

- Optimize API usage by carefully selecting the model parameters, such as the maximum number of tokens. This is necessary to balance the desired output quality with costs.
- When making API requests, be mindful of the rate limits imposed by the platfo Rate limits determine the maximum number of requests you can make within a specific time frame. Exceeding the rate limits may result in API errors or temporary access restrictions.
- Use frameworks and libraries like Langchain to simplify the API interactions. These frameworks offer high-level abstractions and utilities for working with GenAI model APIs.

Building Application using the AI Model

There are several use cases for GenAI-powered applications across various domain

- **Content Creation and Marketing:** GenAI applications can help create outlines articles, ad copy generation, and product descriptions.
- **Customer Support:** AI-powered chatbots can understand user queries and prov accurate, context-aware responses.
- **Business and Finance:** GenAI applications can help generate financial reports, summaries, or analyses based on company data.
- **Education and Learning:** GenAI applications can generate customized learnin material and explanations based on a student's learning style.



Let's say we want to build a chatbot application that uses an LLM to provide personalized book recommendations based on user preferences.

Here are the high-level steps involved.

1 - Choose an LLM Provider

Research and compare different LLM providers, such as Google AI, Open AI, or a Hugging Face.

Before choosing, you can consider multiple factors such as pricing, availability, API documentation, and community support.

2 - Set up the Development Environment

Typically, the LLM providers give access to their LLM via APIs.

You must sign up for an API key from the chosen provider and install the necessary libraries and frameworks.

For example, if you build your application using Python, you should set up a Python project and configure the API credentials according to the best practices.

3 - Design the Chatbot Conversation Flow

Plan out the conversation flow for the book recommendation chatbot. Define the key questions the chatbot will ask users to gather preferences, such as favorite genres, authors, or book themes.

Determine the structure and format of the chatbot's responses, including the recommended books and any additional information to provide.

4 - Implement the Chatbot Application

Use a web framework like Flask or Django to build the chatbot application.

Create a user interface for the chatbot, either as a web page or a messaging interface. Implement the necessary routes and views to handle user interactions and generate chatbot responses.

5 - Integrate the LLM

Most LLM providers have released libraries to talk to their model APIs. Initialize the model with the appropriate parameters, such as the model name, version, and

temperature.

Define the prompts and instructions for the LLM to generate personalized book recommendations based on user preferences.

For example, you can create prompts like: “Recommend a science fiction book for a user who enjoys fast-paced plots and space exploration.”

Pass the user’s preferences and the prompts to the LLM using the API and retrieve generated book recommendations.

6 - Process and Display the Recommendations

Process the LLM-generated book recommendations to extract the relevant information, such as book titles, authors, and descriptions.

Display the recommended books in a clear and visually appealing format. Provide options for users to interact with the recommendations, such as saving them for lat or requesting more details about a specific book.

7 - Refine and Expand

Test the chatbot application with various user preferences and prompts to ensure it generates relevant and diverse book recommendations.

Gather user feedback and iterate on the chatbot’s conversation flow, prompts, and recommendation formatting based on suggestions.

Integrate additional features, such as providing book reviews, suggesting similar authors, and so on, to expand the chatbot's capabilities.

8 - Deploy and Monitor

Deploy the chatbot application to a hosting platform or cloud service provider, mak it accessible to users via a web URL.

Set up monitoring and analytics to track user interactions, chatbot performance, an any errors or issues.

Regularly update the LLM prompts and application logic based on user feedback ar new book releases.

Making Models Your Own

There is significant interest in making models more adaptable and customizable to suit the specific needs of the domain.

Let's look at the main techniques to achieve this goal.

Retrieval-Augmented Generation (RAG)

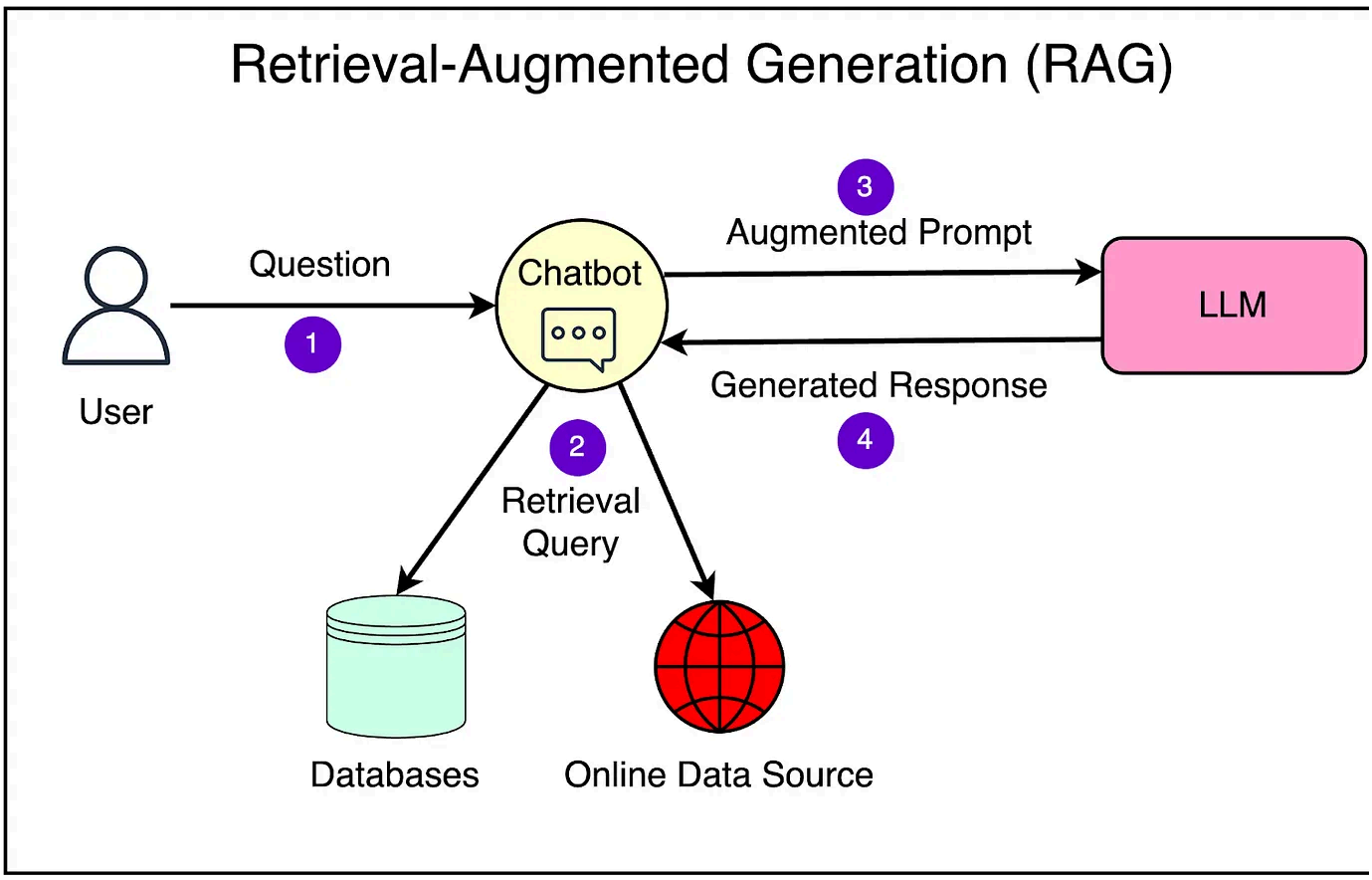
RAG is a technique that helps improve the accuracy and relevance of the generated responses based on your use case.

It allows your LLM to have external information sources like your databases, documents, and even the Internet in real time. This way the LLM can get the most u to-date and relevant information to answer the queries specific to your business.

Here's a high-level overview of how a RAG system works:

- The user poses a question to the RAG system.
- The retrieval component searches the knowledge corpus using the question as query and retrieves the most relevant passages or documents.
- The retrieved passages go through the augmentation step where this informati is fed as input to the large language model. This step is crucial as it augments t model's knowledge with relevant context from external sources.

- The language model processes the input and generates an answer by combinin the information from the retrieved passages and its base knowledge.
- The generated answer is returned to the user.



RAG has shown promising results in improving the accuracy and relevance of generated responses, especially in scenarios where the answer requires synthesizin information from multiple sources. It leverages the strengths of both information retrieval and language generation to provide better answers.

Fine-Tuning AI Models

Fine-tuning a base model on domain-specific data is a powerful technique to impro the performance and accuracy of AI models for specific tasks or industries.

Let's understand how it's done.

1 - Understanding Base Models

Base models, also known as pre-trained models, are AI models that have been train on large, general-purpose datasets.

These models have learned general knowledge and patterns from the training data, making them versatile and applicable to a wide range of tasks.

Examples of base models include Google's BERT and GPT, which have been trained massive amounts of text or image data.

2 - The Need for Fine-Tuning

While base models are powerful, they may not always perform optimally for specific domains or tasks.

The reasons for fine-tuning a foundation model are as follows:

- Adding a specific task (such as code generation or content generation) to the foundation model.
- Generating responses based on your company's proprietary dataset.
- Adapting to the unique vocabularies, writing styles, or data distribution that might differ in your specific use case.
- Reducing hallucination, which is output that is not factually correct or reasona

Fine-tuning allows us to adapt the base model to better understand and generate content specific to a particular domain.

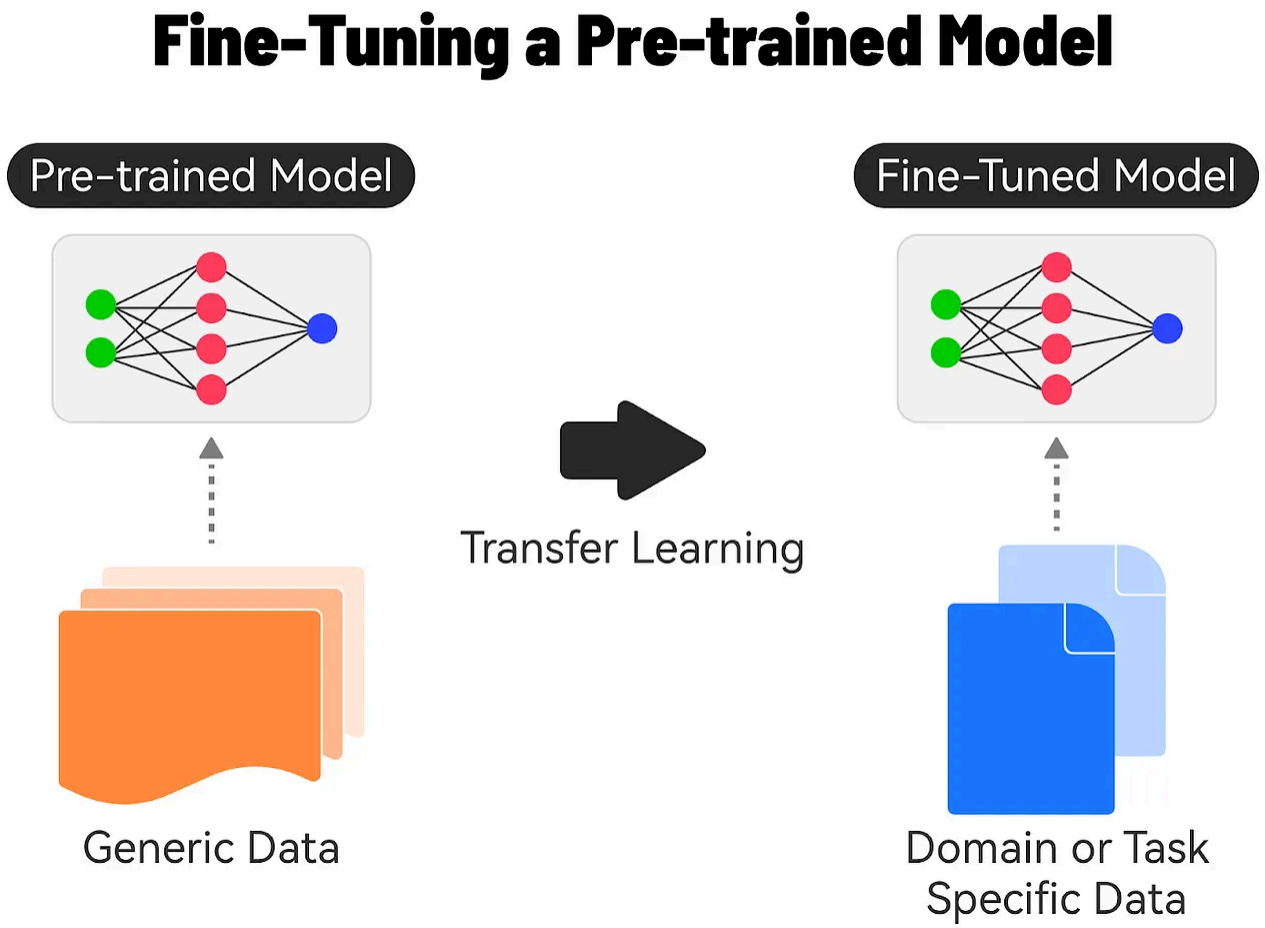
3 - Fine-Tuning Process

The fine-tuning process consists of several steps such as:

- **Data Preparation:** Collect a dataset that is representative of the target domain task while ensuring that it is of sufficient size and quality. Preprocess the data

match the input requirements of the base model.

- **Model Initialization**: Start with the pre-trained base model that is most suitab for the target task. Load the pre-trained weights of the base model.
- **Training:** Feed the domain-specific dataset into the modified base model and train the model using techniques like transfer learning. Fine-tune the model's parameters by backpropagating the errors and updating the weights based on t domain-specific data.
- **Evaluation and Iteration:** Evaluate the fine-tuned model's performance on a validation set from the domain-specific data. Based on the metrics, iterate on t fine-tuning process.



4 - Benefits of Fine-Tuning

There are significant benefits to fine-tuning:

- It allows the model to capture the nuances and characteristics of the target domain, leading to better accuracy and performance on domain-specific tasks.
- Starting with a pre-trained base model, fine-tuning requires less training data a computational resources than training a model from scratch.
- Fine-tuning enables the model to leverage the knowledge learned from the general-purpose training data and adapt it to the specific domain.

Conclusion

In conclusion, getting started with Generative AI is an exciting journey that opens a world of possibilities for developers and businesses alike.

By understanding the key concepts, exploring the available models and APIs, and following best practices, you can harness GenAI's power to build innovative applications and solve complex problems.

Whether you're interested in natural language processing, image generation, or aud synthesis, there are numerous GenAI models and platforms to choose from. You ca create highly accurate and efficient AI solutions tailored to your specific needs by leveraging pre-trained models and fine-tuning them on domain-specific data.

A guest post by

Priyanka Vergadia

Developer Advocate @Google, helping devs learn @GCPCloud Checkout my videos, sketchnotes and other content 👉https://t.co/dvFki6SkLw Opinions = mine