

Generative *AI* Life Cycle



Training /

Building



Data Preparation



Evaluation



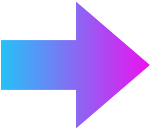
Deployment

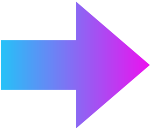


Data Collection















SQL





FEAST



TensorFlow



PyTorch



Feature Engineering



Containers at scale

















Domain Experts



Data Indexing



Context Retrieval



Tool & Function Calling



OPENAI



Hugging Face

ANALYTICAL









OPENAI



Hugging Face

ANALYZING THE





Data Engineer



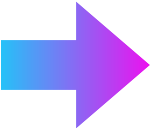
Software Engineer

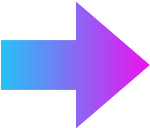


Data Scientist/



M.L.Enginler















Amazon
S3









Method	Definition	Primary use case	Data requirements	Training time	Advantage	Considerations
 Prompt engineering	Crafting specialized prompts to guide LLM behavior	Quick, on-the-fly model guidance	None	None	Fast, cost-effective, no training required	Less control than fine-tuning
 Retrieval augmented generation (RAG)	Combining an LLM with external knowledge retrieval	Dynamic datasets & external knowledge	External knowledge base or database (e.g. vector database)	Moderate (e.g. computing embeddings)	Dynamically updated context, enhanced accuracy	Significantly increases prompt length and inference computation
 Fine-tuning	Adapting a pre-trained LLM to specific datasets or domains	Domain or task specialization	Thousands of domain-specific or instruction examples	Moderate — long (depending on data size)	Granular control, high specialization	Requires labeled data, computational cost
 Pre-training	Training an LLM from scratch	Unique tasks or domain-specific corpora	Large datasets (billions to trillions of tokens)	Long (days to many weeks)	Maximum control, tailored for specific needs	Extremely resource-intensive

How teams are building LLMs



complexy/compute-intensive

2

5