Innover Generative AI Framework – Innsight.AI™

Whitepaper

September, 2023

Author:

- **Ranjith Anantharaman,** Lead Data Scientist, Innover
- **Moitraya Dey,** Lead Analyst, Innover
- **Amit Gautam,** Chief Executive Officer, Innover

Table of Contents

Introduction

Generative AI has garnered immense popularity, captivating the attention of both experts and the public alike. The "Big Bang" moment for generative AI occurred on November 30th, 2022, when OpenAI unveiled ChatGPT, a text generation tool that quickly captured the attention of the masses. This formidable tool astounded users by proficiently crafting diverse forms of content, ranging from blog posts and news stories to marketing emails and even comprehensive articles on a given subject. All it requires is a simple input command entered into a text box. The remarkable success of models like ChatGPT, coupled with the projected market value of generative AI estimated to reach 109 billion USD[1] by 2030 has significantly amplified the excitement surrounding this transformative technology.. In fact, ChatGPT achieved a remarkable feat by acquiring 100 million monthly active users within just two months of its launch, making it the fastest-growing consumer application in history. Its capability to produce text that resembles human language has truly transformed the way we interact with AI-powered systems.

---

[1] https://www.grandviewresearch.com/press-release/global-generative-ai-market

Evolution of Generative AI

The evolution of generative AI spans nine decades, marked by continuous innovation and adaptation. In 1943, Walter Pitts and Warren McCulloch developed Neural Networks, inspired by the cognitive processes of the human brain. These models simulate artificial neurons that respond to stimuli, generate signals with varying strengths, and transmit them to neighboring neurons. Alan Turing's machine intelligence test in 1950 paved the way for Conversational AI, while Noam Chomsky's work in 1957 revolutionized Natural Language Processing by proposing mathematical rules to process grammar.

The 1990s brought about the Long Short-Term Memory (LSTM) model, which addressed earlier neural network limitations. However, it lacked focus on specific input data parts. The introduction of Generative Adversarial Networks (GANs) in 2014 was a pivotal advancement. GANs employ two neural networks, a Generator and a Discriminator, to generate synthetic data and evaluate its authenticity via adversarial training. GANs have allowed breakthroughs in creating photorealistic images and style transfers.

Variational Autoencoders (VAEs) combine autoencoder and probabilistic modeling principles, introducing a probabilistic element into the latent space. This enables diverse data generation and tasks like denoising and anomaly detection. VAEs find applications in various domains, such as style transfer and healthcare.

The attention mechanism, introduced in 2017 by a Google team in their paper "Attention is all you need," has been a game-changer. This mechanism allows neural networks to consider not only their current state but also historical states, enhancing contextual understanding and handling long-range dependencies. Transformer models, relying on attention mechanisms, excel in language translation and text generation tasks. Examples like BERT and GPT have redefined natural language processing by improving sentiment analysis and code generation.

In summary, the evolution of generative AI has witnessed milestones like Neural Networks, GANs, VAEs, and attention mechanisms, leading to significant advancements in language processing, image generation, and diverse data synthesis.

What is LLM and GPT anyway?

Large Language Models are machine learning models that attempt to learn the structure and meaning of language. Large Language Models are designed to analyze and understand natural language queries, known as prompts, and generate the most suitable responses. To achieve this, Large Language Models rely on extensive training with massive language datasets, which helps them gain knowledge and understanding of language patterns. These models are trained on gigabytes of data, making them highly powerful in predicting language. For example - When an LLM tells "the sky is blue", it is confident in that answer not because it has seen or can comprehend what color the sky is, but only because it knows "blue" is the word most likely to follow "the sky is". While LLM refers to a broader framework that incorporates latent variable modeling into language processing, GPT is a specific implementation of a language model based on the transformer architecture. It is pre-trained on a large corpus of text data and can generate human-like text given a prompt. In contrast to the traditional transformer architecture that consists of both an encoder and a decoder, GPT models are autoregressive transformer models that are primarily decoder-only.

This decoder is responsible for both understanding input text and generating output text. It works in a step-by-step manner as below.

1.

Processing Text and Capturing Meaning:
GPT takes text and converts it into mathematical representations called embeddings. These embeddings help in understanding the meaning and relationships between words. Instead of having a separate encoder, GPT directly processes the input in a way that captures its essence.

2.

Context and Positional Information:
GPT uses a technique that considers the position of words in a sentence to understand the context. This allows GPT to comprehend the nuances and differences in meaning even when words are used in different positions. Positional encoding is incorporated directly within the decoder to maintain the order of words.

3.

Generating Output using Autoregression:
GPT generates output in a sequential manner, predicting one word at a time. Each prediction helps determine the next word, ensuring a natural and coherent flow of the generated text.

4.

Attention Mechanisms for Relevance:
Masked Self-attention Mechanisms are employed in the decoder to decide which parts of the input are important for generating each word in the output. Masking is typically done by replacing the words to the right with a special token (often referred to as a "mask" token) or by making those positions unavailable for the model to attend to.

5.

Selecting the Best Output:
GPT estimates multiple possibilities for the next word and selects the most suitable one. It's a bit like a writer considering different ways to continue a sentence and choosing the one that fits best.

In summary, GPT is a single-module (decoder-only) system that reads and understands input text, processes it while considering the word positions, and generates a meaningful and contextually appropriate sequence of words, one step at a time.

Unlike traditional language models that only generate the next word in a sequence, GPT models can generate long and coherent responses. They can take contextual information into account and dynamically attend to different parts of the input.

One of the advantages of the Transformer architecture is its parallelizability. Unlike recurrent neural networks that process words sequentially, Transformers can process the entire input simultaneously, leading to faster computation. Also, transformers, including GPT, can capture long-term dependencies in sequences effectively. The self-attention mechanism allows each position in the input sequence to focus on any other position, enabling the model to establish relationships even between distant words. This ability is essential for understanding and generating coherent and contextually accurate text.

Foundation models and why fine-tuning

While generic LLMs offer a robust starting point, their success in client-specific use cases within the business world is limited to generic use cases. Every organization possesses unique industry knowledge, historical data, and operational nuances that cannot be fully captured by a one-size-fits-all model. Generic LLMs lack the fine-tuned parameters required to comprehend and address the intricacies of individual businesses, leading to suboptimal performance and limited impact on customer experiences. What you gain in speed and simplicity, you lose in control and customization.

We're as of now at a phase in the adoption cycle of generative AI when most organizations are starting to experiment by "off the shelf model" which are known as foundation models. Previously, getting better results from AI was typically a question of “more”—you created more models for more use cases. Foundation models change that. Because they’re trained with internet-scale data, a single model can be adapted and fine-tuned for any number of different use cases. Today’s foundation models are trained on large datasets—and can even be pre-trained without a particular task in mind.

While generic LLMs offer a robust starting point, their success in client-specific use cases within the business world is limited to generic use cases. Every organization possesses unique industry knowledge, historical data, and operational nuances that cannot be fully captured by a one-size-fits-all model. Generic LLMs lack the fine-tuned parameters required to comprehend and address the intricacies of individual businesses, leading to suboptimal performance and limited impact on customer experiences. What you gain in speed and simplicity, you lose in control and customization.

At the other extreme, companies can build their own model, trained with their own data, that’s totally under their control. Creating a proprietary LLM is a demanding and resource-intensive endeavor. It requires substantial investments of time, money, and expertise. Training large models necessitates access to extensive datasets and specialized infrastructure, which can pose significant hurdles for many companies. The costs associated with building and maintaining such models make it an impractical option for most organizations, especially in the initial stages of AI adoption. The training of GPT-3 models was done on 1024 GPUs, took 34 days, and cost $4.6M in computing alone.

Due to the challenges and limitations of creating custom LLMs, **fine-tuning becomes the pivotal step in enhancing the value and impact of AI models.** By fine-tuning, companies can leverage their existing domain data and augment it with the foundation model's capabilities, striking a balance between customization and time efficiency. To increase the value of generative AI and foundation models in specific business use cases, companies will increasingly customize pretrained models by fine-tuning them with their own data—unlocking new performance frontiers. So now it’s about less. You need far fewer models. The emphasis shifts from model building to model fine-tuning and usage. Moreover, Fine-tuning offers greater flexibility for model selection, improvement, and the ability to switch models based on continuous monitoring and evaluation.

In summary:

1. Pre-trained models are good generalists, fine-tuned models are good specialists, so for specialized tasks fine-tuned models excel. Smaller fine-tuned models are cheaper, faster and better at specific tasks.
2. Building proprietary LLMs requires significant investments of time, money, and expertise, making it impractical for many organizations.
3. Fine-tuning becomes crucial in enhancing the value and impact of AI models by leveraging existing domain data with a foundation model's capabilities.
4. Greater control and flexibility to choose models, switch models based on continuous monitoring and evaluations, continuous improvements to the model.

Fine-tuning Methods

Let's delve deeper into fine-tuning, to understand the popular techniques used in fine-tuning.

Fine-tuning is the process of adapting the pre-trained LLM's weights to a domain or knowledge base of interest. Fine-tuning an entire network of a model is expensive and time consuming, also when done with limited data it is highly probable to worsen the results. There is an alternative to fine-tune an entire model referred to as Parameter Efficient Fine-tuning (PEFT), that adds a small number of parameters to the pre-trained model and only the newly added parameters are adjusted leaving the rest of the model's weights untouched.

Broadly there are three popular approaches within PEFT – Adapter tuning, prefix tuning and prompt tuning.

Adapter Tuning:

An adapter is a neural block consisting of a down project of the input representation via a fully connected layer, passed to a non-linear activation, then an upward projection into the original dimension using a second fully connected layer. The adapter is injected inside the transformer blocks of the LLM's. The original connections of the transformer block are preserved via skip connections. The adapter produces an incremental change in the input representation during the feed forward process, which is added back to the original input representation via the skip connections. This allows the adapter to change the behavior of the LLM keeping the original weights unaltered.

Prompt Tuning:

Prompt tuning is a variant of prefix tuning, where soft prompts are learned via backpropagation. Here a prefix vector is prepended to the embeddings of the input tokens, these prefix embeddings are adjusted via backpropagation. The soft prompts learned by the model have been proven to be accurate than the hard prompts / human generated prompts. Since only the prefix embeddings are learnt in the fine-tuning process, prompt tuning is less prone to overfit on the training data, thus more generalizable.

Prefix Tuning:

In prefix tuning, a set of task specific vectors called as Prefix are prepended to the model unlike the adapters that are injected inside the transformer blocks of the model. The prefixes are altered during the fine-tuning process in a way that steers the LLM to generate accurate responses. Prefix tuning plays better when the data at hand to fine-tune is very limited.

Fine-tuning Challenges

In the realm of deploying and utilizing Language Models (LMs), the journey from the inception of a model to a finely tuned, highly performing Inference End Point is not devoid of challenges. While the promise of refined accuracy and domain-specific capabilities beckons, a lurking hurdle obstructs this path—the formidable fine-tuning process. The apparent simplicity of fine-tuning belies the intricate layers of complexity involved, and herein lies the challenge.

Challenging Data Preparation:
Preparing labeled data required for fine-tuning is a significant challenge. Data may reside in various formats like PDFs, requiring sophisticated extraction techniques, including OCR models, to derive meaningful information from unstructured data.

Data Privacy and Security:
Ensuring the privacy of data during the fine-tuning process is paramount. Robust data handling and security protocols must be in place to safeguard sensitive information and comply with privacy regulations.

Hyperparameter Selection Complexity:
Fine-tuning involves choosing from a variety of pre-trained models and employing different techniques, making the selection of optimal hyperparameters a complex task. Brute-forcing combinations are time-consuming and costly, emphasizing the need for a guided approach to streamline this critical process.

Addressing Latency and Model Size:
The sheer size of language models leads to increased latency during inference. To mitigate this, appropriate quantization and model compression techniques must be applied to reduce the model's size and improve response times.

Optimizing Infrastructure Costs:
Establishing the proper infrastructure to support Language Model fine-tuning is costly and challenging to manage. Careful planning, including hardware selection aligned with processing power, memory, and storage needs, and choosing suitable scaling options, is crucial to optimize costs and efficiency.

Continuous Monitoring and Optimization:
Post-deployment, monitoring the model's performance and resource utilization is essential to maintain optimal operations. Continuous evaluation and potential adjustments are necessary to ensure sustained performance and efficient resource usage.

To bridge the gap and triumph over the fine-tuning challenges, organizations must acknowledge and address the inherent challenges associated with fine-tuning Language Models. By developing advanced solutions that streamline the fine-tuning process, automate hyperparameter optimization, enhance model interpretability, and provide comprehensive insights, organizations can unlock the full potential of Language Models, making them more accessible and efficient tools for various applications.

Risks associated with LLM's -

Bias, truthfullness, fairness, robustness.

When using LLMs in policy or technical documents, it's important to be aware of potential pitfalls:

1.0 Bias:
Imagine a customer service script generated by an LLM that consistently recommends higher-priced products to customers of a certain income level. This biased behavior could lead to accusations of discriminatory practices, harming the company's reputation and potentially leading to legal action.

2.0 Truthfulness:
Consider a technical guide generated by an LLM that contains incorrect information about a product's safety features. If someone follows this guide and gets hurt, the company could face legal issues for spreading false information.

3.0 Fairness:
Picture a terms of service agreement that an LLM generates, but it only favors the company and doesn't consider users' rights. This unfair treatment could lead to legal disputes and damage the company's credibility.

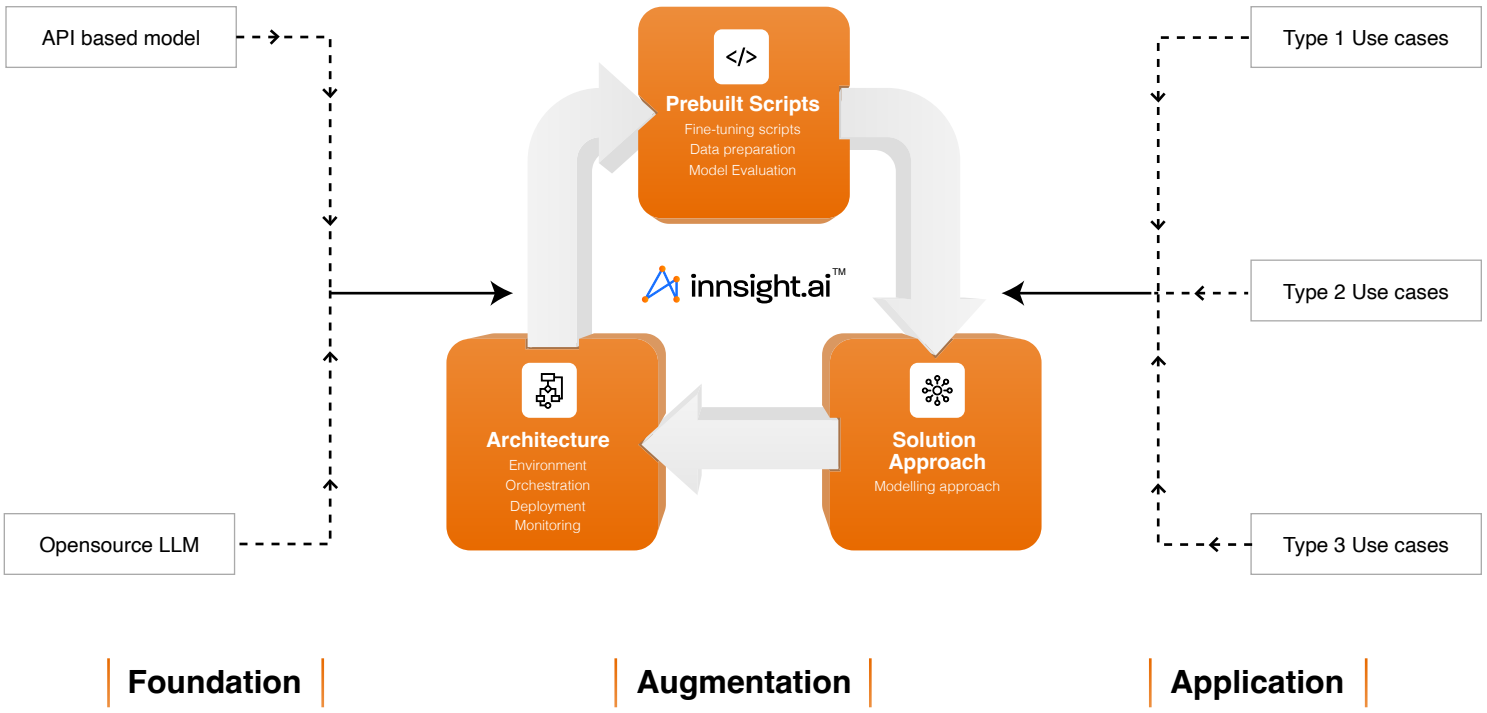
4.0 Robustness:
Think about a troubleshooting manual created by an LLM. If the manual only works when users input very specific phrasing, it might leave many customers confused and frustrated, impacting their experience with the company's product.

To mitigate these risks, it's crucial to carefully review and edit the content generated by LLMs, ensuring accuracy, fairness, and compliance with company policies. This involves a thorough human review process to catch any biased, inaccurate, or unfair information before it reaches customers or stakeholders. By proactively addressing these risks, companies can uphold their values, maintain trust, and ensure that their LLM-generated content aligns with their intended policies and goals. At Innover, we use machine learning techniques to mitigate these risks posed by LLM's, which will be covered in detail in the following whitepapers.

Innover's GPT Framework – Innsight.ai™

While the potential benefits of LLMs are huge, having a robust framework to leverage those benefits while managing the potential hazards and performance expectations is critical. Business's biggest need is speed to value realization in a secure and scalable manner. With the objective of helping our clients leverage the benefits of LLMs for practical business use with context, security, scalability and repeatability, Innover has created its own GPT framework – Innsight.ai™.

This framework offers insights into architectural principles, the LLM stack, and key conceptual considerations at critical junctures throughout the development process of LLM-based applications.



Innisght.ai framework offers guiding principles or key considerations to be addressed while developing an LLM solution. The framework is divided into 3 borad components and their sub-components.

- Foundation or Model Selection
- Augmentation
  - Solution Approach
  - Architecture
  - Prebuilt scripts
- Application or Use Cases

Lets dive deeper into each aspect of the framework.

1. Foundation

Right model selection is probably the most important consideration in solving the business problem. Hence all the factors mentioned here are also part of the Solution aspects in the Augmentation layer. The following factors are considered as part of the decision making.

1.

Accuracy matters:

Closed source off the shelf commercial models excel in comprehending prompts, rendering more effective instructions interpretation. This type of model usage is referred to as in-context learning, where a prompt is constructed with an input question, a context, and an instruction, and the model generates an answer. There is another usage of the model referred to as fine-tuning, where a pre-trained model is adapted to a specific task or domain. The choice of a modelling approach between in-context learning and fine-tuning depends on various factors which are discussed in the further sections, the basic premise is that fine-tuning will yield accurate results for specialized tasks under the availability of sufficient quality data, prompting works better as a generalist and can yield comparable accuracy if a pre-trained LLM was already familiar with the domain at hand.

2.

Number of requests:

The expense of incorporating closed-source models can vary, depending on the number of requests produced by the application. In cases of a high volume of requests (1 million+), it's usually more cost-effective to utilize open-source LLMs. Infrastructure costs of serving the open source models generally turn out to be cheaper in the long run in case of high volume of requests. However the costs of infrastructure for serving open source models and api subscription costs for commercial models rapidly change, so it is best to arrive at a cost calculation template dynamically.

3.

Data Privacy:

Off-the-shelf applications typically require data to pass through the app provider's cloud. A custom-built application can remain in an enterprise's virtual private cloud - or even on-premises - for cases where data security is critical. With a purpose-built application, data stays within the existing environment, so access control of any deployed LLM app can mirror existing role-based access controls.

A primary initial concern revolves around safeguarding the privacy of customer-shared data. When dealing with closed-source LLMs, data is transmitted over the internet to their servers for inference purposes. It is vital to guarantee that this data remains secure and is not utilized for training. It is imperative to secure customer consent regarding data usage policies.

The aforementioned consideration yields two potential outcomes: either the customer consents to utilizing closed source commercial models or withholds approval. In the event of approval, the exploration of open-source vs closed source LLMs becomes essential for subsequent solution development.

For cases where organization’s security policies restrict the passage of data to the provider’s cloud, a custom built application that engages an open source LLM is the only path ahead.

2. Augmentation

2.1 Solution

When developing a solution for LLM application, there's a choice between two ways of building a model: In-Context Learning (or Prompting) vs Fine-tuning. Also these two approaches need not always be implemented in silos, they can be combined as well, fine-tune a model to adapt to the domain, and use prompting on the fine-tuned model to infer for the specific downstream task. Below are some of the scenarios where each technique works best.

2.1.1
Meaning and Purpose

Fine-tuning: Tailors models for specific tasks by adjusting pre-trained parameters.
In-Context Learning: Improves performance through task-specific instructions incorporated into the learning process.

2.1.2
Data Size and Quality

Fine-tuning: Effective when data size or quality is high; works well with a substantial number of samples (question-answer pairs). A pre-trained model is to be chosen first, choose one that has been pre-trained on relevant domain's data. Once a pre-trained model is chosen, high quality data is to be curated either manually or by using generative models in turn to prepare the data. Too few data samples would not make significant changes to the pre-trained model weights, neither does too much data without diverse prompts.
In-Context Learning: Suitable when the data's quality and quantity support the integration of task-specific instructions.

2.1.3
Context Consideration

Fine-tuning: Fine-tuning is the process of making a pre-trained model better at specific tasks. It's really good at understanding and using context, especially when questions and answers are spread out over a lot of information. Fine-tuned models can adapt and take in a lot of context, which helps them give accurate answers even when the context is quite extensive. So, they excel at handling complex and detailed information.
In-Context Learning: In-context learning, on the other hand, relies on carefully crafted instructions (prompts). It's efficient when the context can be summarized in short prompts. But it might struggle when dealing with vast amounts of information or when the context stretches across multiple paragraphs. In such cases, it can be tough to fit all the necessary context into short prompts, which may result in less accurate responses when compared to fine-tuned models.

2.1.4
Inference Latency

Inference latency is the time it takes to generate a response during runtime.
Fine-tuning: Faster inference at runtime, as the model is optimized for the specific task. We might fine-tune smaller models pre-trained on relevant domain for specialized tasks, thus it is lighter than the huge out of the box commercial LLM's.
In-Context Learning: Might introduce slightly higher inference latency due to the broader approach. Deriving the relevant context to be fed to the prompt, along with inference on an out of the box bigger pre-trained LLM incurs time.

2.1.5
Data Updates

Fine-tuning: Can become challenging when dealing with rapidly changing, live data. Monitoring a model for performance degradation, fine-tuning the model on changing data periodically incurs cost and effort.
In-Context Learning: More adaptable to changes in data, as the task-specific instructions can be updated with ease dynamically.

2.1.6
Localization and Customer Level:

Fine-tuning: More difficult to build fine-tuned models at a highly localized customer level.
In-Context Learning: Offers flexibility to incorporate knowledge base and instructions for specific customer needs.

2.1.7
Use case dependent

Consider the nature of your use cases.

2.1.7.1 If fine-tuning seems impractical due to data availability, in-context learning might be more suitable.

2.1.7.2 For tasks that have objective responses fine-tuning on smaller models generally might be better suited, because the fine-tuning process involves learning from a curated labelled dataset that has an objectively true answer per instruction and context, fine-tuned model would have well captured the relation between the instruction and label good enough, whereas it is not possible to accommodate the variety of relationships between instruction and label in a prompt when used in in-context learning.

2.1.7.3 If the application being built is envisioned to serve multiple knowledge bases or customers, in-context learning is scalable. Fine-tuning works well when the objective is to build accurate model for a specific task and data.

In-context learning and fine-tuning present their respective advantages and disadvantages, with relevance contingent upon the specific scenarios outlined earlier. This paper emphasizes fine-tuning, reserving an in-depth exploration of in-context learning for forthcoming publications.

2.2 Architecture

Running a fine-tuning experiment on a knowledge base involves careful setup of technical architectural components. Innover's GPT framework enables running an LLM fine-tuning experiment by providing the necessary setup.

2.2.1
Decoupled Environment

In our approach, we aim to optimize how we handle the environment in LLM development pipeline. Instead of burdening them with environment management, we propose creating a tailored environment independent of the training scripts. This custom environment can be packaged into a Docker container and sent to a designated computing target.
This strategy offers a significant advantage—it facilitates seamless sharing of the environment among team members, promoting smoother collaboration. Team members can work without getting bogged down by complex setup details, keeping the training script clean and easy to handle.

2.2.2
Model Orchestration

Our orchestration pipeline involves scripts for fine-tuning language models (LLMs), evaluating generative responses, managing model registry, and deployment. When fine-tuning an LLM, we add a small number of learnable parameters to the pre-trained model, during registration of a model we focus on registering only the smaller adapter weights. The pipeline consists of a scoring script that makes use of the registered model, pre-trained LLM and tailored environment to generate the response.

2.2.3
Deployment

Deployment-wise, we opt for fully managed hosting services provided by major cloud platforms instead of self-managing and hosting the solution. This choice is driven by the built-in security features, automatic scaling, monitoring and faster time to market capabilities, offered by these cloud providers.

2.3 Pre-built scripts

The framework comes with pre-built scripts for various essential functionalities, including:

1
Data Pipelines:

- Ingestion of data
- PII screening
- Labeled dataset generation from unstructured documents
- Secure data storage

2
Orchestration Pipeline:

- Fine-tuning scripts capable of fine-tuning on various open-source language models (LLMs)
- Model-agnostic evaluation engine for generative responses

3
Highly Customizable Applications:

- Developed using modern web frameworks
- Allow for easy customization and immediate utilization of GenAI efforts

Using the framework we were able to develop LLM based applications at speed, following are some of the interesting apps developed by Innover.

3. Application or Use Cases

Generative AI has gained significant recognition in the business world, although its usage in solving business use cases is still in its early stages that in the coming years, generative AI will be increasingly adopted by businesses to address various challenges and boost productivity significantly. Adoption of GenerativeAI solutions will vary depending on business complexity as well as integration with other evolving technology solutions like RPA, voice assistants, etc.

For simplicity, Innover has broken down potential use cases into three broad categories based on ease of application and multiple technologies integration complexities.

Type 1: On-Demand Information availability:

Extracting useful information in dynamic and time bound fashion for business decision making will be the easiest and quickest to adopt. We are witnessing significant demand and adoption for this Type 1 business use cases. By using generative AI models, businesses can develop question answering chatbots and intelligent control towers that provide self-service capabilities to stakeholders. These solutions aim to standardize processes and reduce the time spent by subject matter experts on repetitive activities.

While business is still experimenting with different use cases, tech industry is an early adopter resulting in meaningful productivity boost, especially for easy simple and medium complex software engineering. Rapidly converting one programming language to another, mastering programming tools and methods, automating code writing, predicting and pre-empting problems and managing system documentation, these are some of the examples by which software professionals have kick started their journey with generative AI.

Type 2: Hyper-personalized end to end transactions in real time :

While machine learning models have successfully driven personalization that businesses have leveraged for driving better experiences for its customers, employees and suppliers, they have largely been leveraged for segments of the entire transaction. Generative AI, along with Machine Learning models promises to drive better personalization, information availability in real time and enable the entire engagement experiences for customers, suppliers and employees. We are actively engaged in use cases around service desk, field services, supply chain visibility, marketing and eCommerce where Generative AI is leveraged for providing end to end transaction experience improvement.

Type 3: Intelligent ecosystem & Independent Decision Making:

The ultimate maturity of generative AI will be reached when it will get combined with technologies such as voice assistant, speech to text and robotic process automation and will create real time impact by acting as an intelligent and independent decision making system. It will be able to automate end to end transaction with customers and employees. With its ability to autonomously generate meaningful actions and make decisions based on value, risk, and likelihood, it can and enhance business processes. For instance, a generative AI tool capable of understanding the need and sentiment of customer during conversation will be able to solve their problems, sell products and provide discount to disgruntled customer. Similarly, it will be able to summarize financial reports for an entire industry and making reliable investment decisions could save time and improve investment strategies. Overall, the integration of generative AI and other technologies has the potential to provide valuable information and make informed decisions with minimal human interaction, thus driving productivity to new heights.

We trust that this whitepaper has been insightful and relevant to your needs. It aims to deliver a notable positive impact on the business world by leveraging the capabilities of Generative AI and Language Models (LLMs). Our framework serves as a stepping stone towards showcasing the value we can bring to the business landscape.

Authors:

Ranjith Anantharaman, Lead Data Scientist, Innover

Ranjith holds the position of Lead Data Scientist at Innover, bringing with him a wealth of experience spanning 13 years in this dynamic field. His expertise encompasses a broad spectrum of cutting-edge technologies and methodologies, including Large Language Models, Natural Language Processing (NLP), Customer Analytics, Text Analytics, Predictive Analytics, and the intricate realm of Big Data Analytics.

Ranjith's passion lies in harnessing the power of data to demystify complex concepts and drive data-driven decision-making for client success. He has spearheaded numerous transformative projects, leveraging his deep knowledge to deliver actionable insights and measurable results.

Moitraya Dey, Lead Analyst, Innover

Moitraya Dey is the lead analyst at Innover. Moitraya brings a decade of experience where his proficiency in data analytics, artificial intelligence, and comprehensive knowledge of the industry landscape have consistently yielded positive results for global brands. Moitraya's unwavering commitment to excellence, genuine zeal for turning data into strategic decisions, and a keen eye for emerging technology trends have helped him spearhead transformative projects that not only optimize operational efficiency but also foster innovation and sustainable growth for the customers.

Amit Gautam, Chief Executive Officer, Innover

In his role as Chief Executive Officer for Innover, Amit Gautam has responsibility for all aspects of the company's product & services strategy & execution, as well as its financial performance and growth.

Amit has a relentless focus on growth and innovation and holds a strong personal commitment towards "outcome-driven" digital transformation for businesses. Amit collaborates with the C-suite executives of Fortune 1000 companies and guides them to adopt a digital-first mindset, delivering bold transformations and exceptional experiences. Prior to Innover, Amit worked with firms like GE and Cognizant in various leadership roles.