Getting Beyond the Sandbox: A Playbook for Developing Generative AI Solutions for Enterprise Applications

Jay Mandal and Dr. Megan Ma

Introduction

Companies are captivated by the potential of generative artificial intelligence (AI) to transform their businesses, but many don't know exactly how. Generative AI is defined as artificial intelligence capable of generating text, images, video or other data using generative models. Companies in all fields are experimenting in sandbox environments of different ways this emerging technology can provide value to customers. This urgency is also partly owed to leadership and market pressures to release new generative AI-driven solutions. The main questions for companies are asking: (1) where should be the focus; and (2) how do we get started?

This is a generative AI playbook that provides companies a framework on how to prioritize use cases and start building. The key elements covered are:

1. Developing solid use cases;
2. Developing a solution and using proprietary data;
3. Which type of generative AI model to use;
4. Risk mitigation to enable enterprise readiness for generative AI; and
5. Optimizing the user-experience (UX); and
6. Conclusion

The focus of this paper will be on generative AI solutions that benefit external customers, and not solutions that benefit employees and improve internal company processes.

I. Developing Solid Use Cases

The following product discovery process can help a company identify and build generative AI solutions that provide the most value to their customer base. To start, a company should not rush to build a generative AI solution in search of a problem. Instead, an enterprise should first prioritize key customer pain points and then determine where generative AI driven solutions could best address those key customer problems.

First, focus on the user problem. A company should acquire a deep understanding of problems for a subset of customers, and then prioritize these problems. In order to do this, a company should begin with a user problem discovery process. For example, the team would research, interview, or collect data on a subset of customers that share common characteristics (a user persona) with the intention of identifying the user persona's main pain points. A user persona is important because the team should not just identify broad problems in the space the company

operates, but identify acute pain points for a subset of customers to determine the true needs of the market. For example, a user persona could be defined as startups based in technology hubs like Silicon Valley only at the seed stage or earlier of funding. Following this assessment, the company would prioritize the user persona's pain points. For example, the team could assign a value of high, medium, and low to each of these pains felt by a user persona.

Next, develop generative AI solutions to address top pain points. For one or a couple of the most pressing pain points of a defined user persona, the team would then brainstorm solutions incorporating generative AI. The aim would be to significantly improve a pre-existing solution for the user persona by as much as 10x better in value or cost. Section II will enumerate a few ways to develop generative AI-driven solutions. As well, Section V will explore how a differentiated user experience – defined by how a customer navigates the solution – could also provide value to the user persona.

It is important to assess the necessity of generative AI models in these solutions. If a solution can be accomplished more efficiently without generative AI, then there is no need to use this technology. Also, the generative AI models must also be well suited for the task at hand and technically feasible. For example, a team can rank each solution as high, medium or low (or on a corresponding 1 to 5 numeric scale) in the categories of: (1) how effectively the solution will solve the user persona's problem; and (2) how technically difficult it will be to deliver this overall solution including generative AI. These two rankings would provide guidance to the team on which solution(s) would be the best to pursue for their client-base.

Also, it is certainly not a hard and fast rule that the proposed generative AI solution must be as much as a 10x cost or value improvement for a customer. A company must consider the stickiness or switching costs of a current solution used by a customer – whether it's their own, their competitors, or a homegrown solution. For example, if the switching costs from a competitor are low or the customer has already adopted a prior version of the enterprise's solution (perhaps without generative AI), the company only needs to create a slightly better generative AI solution to win over the customer. This analysis will determine how much better a new generative AI-driven solution will need to be to displace a customer's current solution.

II. Developing a Solution and Using Proprietary Training Data

Types of generative AI solutions

A general framework on the different generative AI-driven solutions, organized on the human-machine collaboration spectrum, is as follows:

1. Co-pilots: that assist human with ongoing tasks, in which a human must initiate prompts to a model when needed to complete specific tasks
2. Bot with human exception handling (e.g. supervisory AI agents): the AI will handle simpler tasks independently, and either ask for human intervention to approve how it handled such a task, or ask for human intervention on more complex tasks, or to handle exceptional cases.

3. Autonomous AI agents (e.g. autonomous co-workers): the AI will carry out simple and complex series of tasks autonomously, without human intervention.

Co-pilots are currently the most commonly created and used solution, next to chatbots. This will be the focus of this paper on solutions companies should create in the near term and mid-term. For example, there are co-pilots for software development, sales processes, marketing content, legal work product, medical diagnosis work, and more. Solutions that fall in category two and three are more experimental. However, there are developments with generative AI models and innovations by early-stage startups that are accelerating their commercial viability.

Autonomous generative AI agents is a recently emerging area. For example, Cognition's product Devin is an autonomous software engineer released for customer beta testing and for demo purposes. In its demo, Devin is provided abstract goals and would independently reason on paths to achieving such goals, including how to complete projects with software-related tasks and challenges. More product experiments and research are needed in this space to determine how effectively an agent can take actions to fully complete projects, but these are early symptoms towards this type of engagement and collaboration between humans and machines.

Prompt engineering and incorporation of proprietary data to customize generative AI outputs

When developing a generative AI solution like a copilot, a company can simply ask or guide the user to input specific instructions to direct a generative AI model toward desired outputs (otherwise known as prompt engineering). Useful guidelines on how to optimize prompt engineering are provided by the teams developing ChatGPT and Claude. However, simply using prompt engineering techniques in a generative AI solution to provide desired outputs to a customer could be easily imitated by a competitor.

Alternatively, a company could create a customer solution with differentiated outputs by augmenting prompt engineering techniques with unique proprietary data added to query or model. This could be customer data, a company's own research database, or other relevant and unique data, all of which was not originally included in the pre-training for a generative AI model. This type of solution would create unique outputs – in terms of content and insights, or structure or style of response – based on the content of the proprietary data. It would also help better curate responses, mitigating risks like hallucination. Nevertheless, prompt engineering has varied performance with mitigating against other risks, such as inconsistency and completeness of responses, nor do they ensure the right safeguards are in place for sensitive data. (See Section IV for a deeper dive on mitigating risks). In any event, many companies are experimenting with solutions that draw from their unique data sets.

Here are methods to use proprietary data to customize generative AI solution outputs:

- ***Context window*** is the input text that a generative AI model can evaluate for response generation. For the latest models, the size of the context window has increased so much

that proprietary data could be included in the prompt context window preceding a query. For example, the context window of GPT-4 currently ranges from 8,000 to 128,000 tokens. The largest Gemini 1.5 Pro model (currently under limited release) is up to 1,000,000 tokens. This is the rough equivalent of about 700,000 words (the size of multiple books), 11 hours of audio, or one hour of video. However, as the context window becomes longer, or as a user needs to provide more context, the generative AI model can begin to forget this data (see for example results from this evaluation). Furthermore, it can get rather costly to run the model. In these cases, an alternative solution, retrieval-augmented generation, may be a better option.

- ***Retrieval-augmented generation (RAG)*** is the method by which a model is constrained to reference data outside the model's pre-training data, such as a company's proprietary database, when asked to respond to a query. In effect, the RAG method is like searching for the relevant parts of the outside reference data during a prompt engineering query, and automatically inserting those parts into a prompt alongside the query. RAG is also less expensive to use than inputting all the data into a context window. Instead, RAG inserts only the semantically matched data into the context window.

- ***Fine-tuning*** a pre-existing model, in which a model is further pre-trained on additional specific proprietary or non-proprietary customer data. Prompt engineering queries on a fine-tuned model automatically result in outputs that are unique in terms of content, style, or even structure of answers based on the way it was fine-tuned.

- ***Pre-train a model from scratch with your own data***. Specifically, a new model could be created by pre-training it with proprietary data and other specifically earmarked data relevant to customer use cases. This model would be a purpose-built solution to provide responses based on the newly trained data. However, this is the option of last choice as it is often prohibitively expensive (reaching up to the tens of millions of dollars to create a robust model) and time-consuming for a team to pre-train and create this new model.

Data readiness and paths to get to market?

Data readiness for proprietary data is often a major bottleneck that impacts a company's ability to leverage generative AI solutions. When dealing with unstructured text (such as text-based long form content), such data can more easily be preprocessed by your company's engineering team and then ingested for use by a generative AI model based on the methods above. However, it is more complex if the proprietary data is in the form of various structured databases (such as in table form) disseminated across multiple formats and housed in different centers across a company. Before a model can use this data across various structured databases, the company's engineering, artificial intelligence/machine learning (AI/ML), and data science teams would need to (1) create a semantic layer which translates all such data in a way a generative AI model could use it, and which recognizes and reconciles any underlying redundancies, discrepancies or other ambiguities in the data, and (2) if needed, further prepare and cleanse the data before it can be properly understood by this semantic layer. This may be a time-consuming and involved process for a company's engineering, AI/ML, and data science teams to prepare and organize accordingly.

How to quickly get to market with unique data sets based on readiness.
A company could focus on quick wins by determining use cases that only require training their models on readily available subsets of proprietary data (structured or unstructured). While this may be a much smaller subset of the data a company would like to use, it allows the product to arrive in the hands of customers for use, feedback, and quick iterations, enabling client retention. This data could be used in models with context windows, RAG or even fine-tuning methods to provide customers unique value-add outputs.

A mid-term strategy would be for the company to create a more robust generative AI solution by augmenting it with a much more complete dataset of the disaggregated structured and unstructured data, using the methods mentioned above. A company could also create a more data-centric culture within the company in order to collect more robust and complete data sets to use with these models. This new culture could include requiring current and future products to collect more data (content, text, audio, etc.), transferring data in a more central repository, and maybe even centralizing the prioritization and support process for development of generative AI solutions by a company.

III. Which Type of Generative AI model to Use

When developing a new generative AI solution, a decision facing companies is whether to use: (1) proprietary models (GPT, Claude, Gemini, etc.) or (2) open source models (such as from Llama, Mistral, DBRX, etc.).

The benefits of using a proprietary model are that they are well-suited for general purpose use, as they rely on large, robust models that are highly performant in a wide array of fields. Each of these proprietary model companies also offer smaller models, which are less expensive to use, have lower latency, and are sufficient for more narrowly specified tasks. It's also easier for a company to use proprietary models since they are easier to deploy "out of the box" for a company's solutions. The pricing structure is also reasonable, especially considering their infrastructure encourages a plug-and-play approach. The models themselves are also incredibly capable without much customization. However, there may be vendor lock-in and data privacy concerns about using proprietary data in such models.

The benefit of using open-source generative AI models is that a company has greater flexibility, including full control of the usage, customization, and hosting of the model. There are relatively limited data and privacy concerns with a company's proprietary data as the model, underlying data, and its usage (in context windows, RAG, fine-tuning, or other methods as described above) can be housed as a local instance on enterprise devices. In effect, the data would not leave the company ecosystem. The downside is that it is more time-consuming and costly upfront to set up the model if used in a custom manner.

An approach which companies are adopting is to first experiment with a solution on a premier proprietary model (such as ChatGPT, Claude, Gemini). Once a solid use case is identified, experiment with that same solution on other open source models, or other proprietary generative AI models to optimize for other factors such as price, performance, ease-of-use – so long as data/privacy concerns with underlying data are mitigated. Eventually, the enterprise may migrate the solution to an open source model to have complete control of the underlying model, data, form factor, etc. for the reasons mentioned above. In the very rare case, a company may choose to pre-train their own LLM with proprietary data and even hand-picked public data. However, this last approach may be prohibitively expensive, more time-consuming, and technically more demanding.

IV. Risk Mitigation to Enable Enterprise Readiness for Generative AI

Companies have lower risk appetites in the use of generative AI, in light of their known limitations. These include hallucinations, inconsistencies in response, and most recently, completeness. Contractual and regulatory compliance is another key consideration. These limitations need to be addressed before a model transitions from experimental and exploratory (sandboxing) to production-ready application.

In order to ensure enterprise preparedness, that generative AI tools perform the tasks envisioned for their purpose, risk mitigating measures must be taken at every stage of the development cycle. This means that both defensive and preventative methods must be taken. We consider first defensive techniques, specifically through the lens of security. This includes red-teaming, and the use of adversarial attacks, such as prompt injections, to help identify relevant areas of vulnerability in the existing use of the application. Red-teaming is a common method of intentionally crash-testing generative AI models and has been touted as a form of AI auditing.

Once vulnerabilities (known risks) have been better identified, it may be important to understand how to curtail outputs of the model, such that they are consistent with the expectations of relevant stakeholders. There are a few options that may be helpful in this space.

Guardrails are a set of machine-learning models or rules used to validate the output of the large language model (LLM). They detect whether there is the presence of a specific type of risk. Guardrails AI, for example, is a Python library that allows users to add structure, type, and quality guarantees to the outputs of generative AI models. This means that users may specify and validate how a generative AI model should behave and take remedial action when the model behaves outside of these predetermined rules. This includes, for example, preventing security vulnerabilities by verifying that responses do not contain trade secrets and/or personal identifiable information (PII).

GAI plays a vital role in facilitating conversations among stakeholders and establishing realistic expectations for our problem-solving process. This approach involves actively engaging stakeholders in defining and implementing appropriate safeguards to mitigate potential harms caused by generative AI model applications. Furthermore, GAI recently released Guardrails Hub,

a repository that is open-source and available to ensure against stakeholder-specific risk at the enterprise level.

Similarly, NeMo Guardrails, an open-source LLM security toolset released by NVIDIA, offers a broader set of programmable guardrails to control and guide LLM inputs and outputs. These include content moderation, topic guidance, which steers conversations towards specific topics, hallucination prevention, and reduces the generation of factually incorrect or nonsensical content, and response shaping.

Equally, LlamaGuard offers a viable solution to address the generative AI model input-output vulnerabilities and combat prompt injection. LlamaGuard is effectively an LLM, fine-tuned from Llama-2, that generates text determining whether an input (prompt) or output (response) is deemed safe. It is defined on a taxonomy of six unsafe categories, and developers can have the option of customizing those categories by adding additional unsafe ones to tailor to their specific use cases.

An important step involved in deployment requires processes that enable the continuous monitoring and evaluation of generative AI performance. This applies to both proprietary "out-of-box" models as well as for customized and/or open-source models that have been fine-tuned with enterprise's proprietary data.

Arize Phoenix, another open-source Python library, helps evaluate generative AI model responses and catches issues once an application is deployed in production. That is, Phoenix allows for LLM observability, defined as having total visibility into every layer of an LLM-based software system: the application, the LLM prompt and the LLM response. In effect, Phoenix can visualize datasets and troubleshoot common issues of generative AI like hallucination, relevance and accuracy of responses, quality of summarization, and more.

Together, these toolkits behave as tangible, forensic safeguards that provide more behavioral certainty on ways to effectuate enterprise readiness for production-level deployment of generative AI tools.

A final consideration to address are legal and policy guidelines that work in tandem with aforementioned technical interventions. For example, if the use of the proprietary data includes sensitive customer information, certain clients may have contractual limitations as to the extent to which their data could be used in relation to frontier technologies such as generative AI solutions. Furthermore, there are evolving federal and state regulations with respect to a company's usage of customer data and data privacy, such as with PII (i.e., GDPR). A company should involve a legal team to collaborate on considerations with usage of data, and to navigate emerging areas of regulatory compliance on the usage of generative AI models.

V. Optimizing the User Experience (UX)

Investment in generative AI user experience has been comparatively minimal in comparison to investment in the generative AI technologies (training, inference, mitigating risk, tools/infrastructure, etc.). User experience (UX) may be defined as how easily a user can navigate a product and find what they need. The UX for generative AI models have not moved much further beyond text-based chat bots. Companies need to think carefully about the UX of their generative AI solutions to ensure that customers can best avail the benefits of these technologies.

There are currently major developments for generative AI models in the following areas that require a rethinking of the UX in order to best deliver these solutions to customers:

1) Multimodal capabilities – using not only text, but voice, vision, sensory inputs and other media, as both inputs and outputs
2) Integration of generative AI models – through APIs or homegrown models – into different steps of an enterprise workflow for customers
3) No code/low code approaches to create and use generative AI models in different subject areas (such as Open AI's GPTs, which can be created by users without technical skill sets, or Rivet from Ironclad)
4) Development of agentic generative AI solutions, which autonomously complete entire series of tasks without human intervention

One area of consideration is voice-based inputs and outputs for a model, which are now being offered in many generative AI models. A client may find it better to interact with a company's model by voice dialogue, instead of text-based interaction. Additionally, a company could contemplate a simplified voice-based model instruction interface to guide a model. This would be in place of complex prompt engineering techniques currently being used to optimize outputs from a model. The user could converse with the model in natural language and in the process the model would capture enough relevant instruction to achieve the user's intended outputs. For example, a voice AI that would act as a customer service chatbot that you could converse with in order to solve a product issue (e.g. Vapi).

In our recent paper "Redefining UX Design for Generative AI Models for Enterprise," we identify and explore future user experience paradigms we expect to be offered in models in the near future.

Companies can follow UX research and design thinking methodologies to determine the best UX delivery mechanisms to customers. Some key steps to consider in this UX discovery process would be:

1) Assess customer needs through user interviews with both existing clients and relevant stakeholders;
2) Explore and create designs of the solution (without the underlying technology) that meet those needs;
3) Test those designs and the workflow with a customer;

4) Capture feedback through pilot testing; and
5) Continuous iteration on UX designs before transitioning from prototype to product.

The Stanford D. School and others offer more detailed examples of UX design methodology that companies can follow.

VI. Conclusion

This playbook lays out a practical framework for enterprises to follow as they move from experimentation to production-ready generative AI solutions. A company's leader(s) in this space should be well-versed on the key steps provided in this playbook. This will help them develop a plan to smoothly orchestrate the development and deployment of production-level solutions in coordination with key stakeholders within and outside the company (such as product, engineering, AI/ML engineering, UX team, legal, sales/marketing, outside generative AI model/tools/support companies, and customers). Future questions to explore are:

1) Which verticals are best positioned to make the transition now to developing production-ready solutions?
2) Which emergent technologies in generative AI are poised to be future areas of emphasis in enterprise solutions? Could they be generative AI agentic frameworks, unique UX delivery mechanisms, or something else?