Fivetran

A practitioner's guide to data integration for AI

Practical, step-by-step guidance for centralizing AI-ready data

Table of contents

AI projects are both critical and difficult

Artificial intelligence offers far greater capabilities than conventional reporting and business intelligence (BI), enabling not only better decision support but also predictive modeling, adaptive products, and automation of all kinds. It is a power tool for the human mind, with the potential to accelerate every form of creative and intellectual work.

Despite the promise of "Big Data," AI projects routinely fail to gain traction. About 80% of data science projects, including machine learning (ML) and AI, never make it into production. A recent survey by Bain found that, in a 5-month period spanning late 2023 and early 2024, rates of production implementations of generative AI (GenAI) shrank even as pilot programs grew.

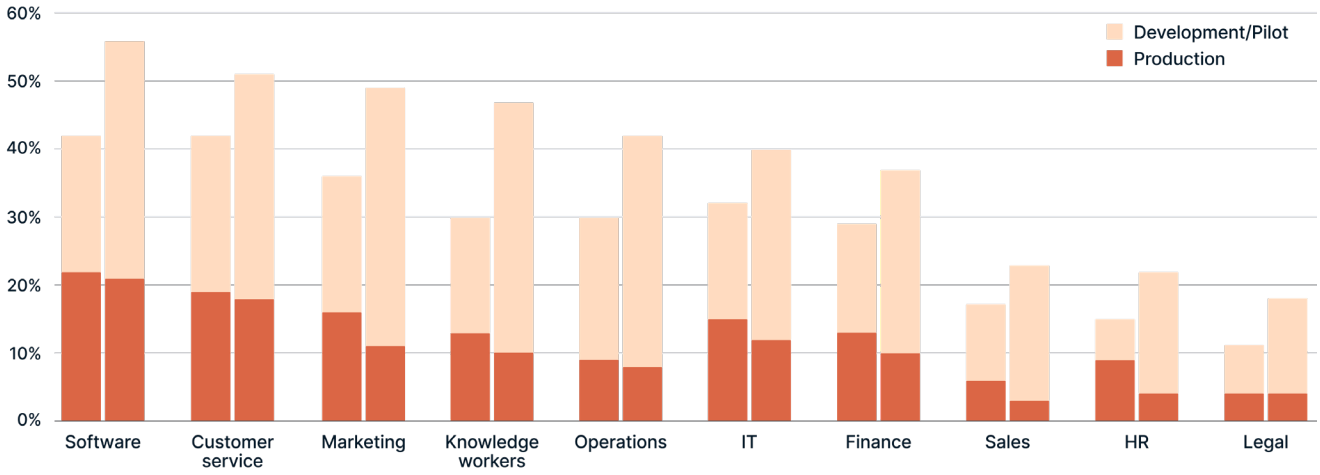
AI projects fail because data teams struggle to establish a modern data foundation capable of supporting modern data use cases. Analytics of all kinds, ranging from reporting and BI to predictive modeling and GenAI, depend on centralized access to a reliable, timely supply of data from a wide range of sources. Successful AI projects require a systematic approach to moving, transforming, and governing data.

This introductory guide provides the essential capabilities, processes, and tools data professionals need to successfully execute AI projects.

Huge interest, limited use so far

Enterprise software takes time, and come with early disappointments

Enterprise use case adoption rates for generative AI, October 2023 & February 2024



Source: Bain

Traditional, generative, and agentic AI

It is important to distinguish between traditional, generative, and agentic AI. Before the release of ChatGPT in late 2022, "AI" commonly referred to non-generative models used for numerical and categorical prediction. Non-generative AI use cases range from extensions of reporting and BI, such as forecasting and pattern recognition, to automated agents using reinforcement learning, such as game-playing bots and self-driving vehicles.

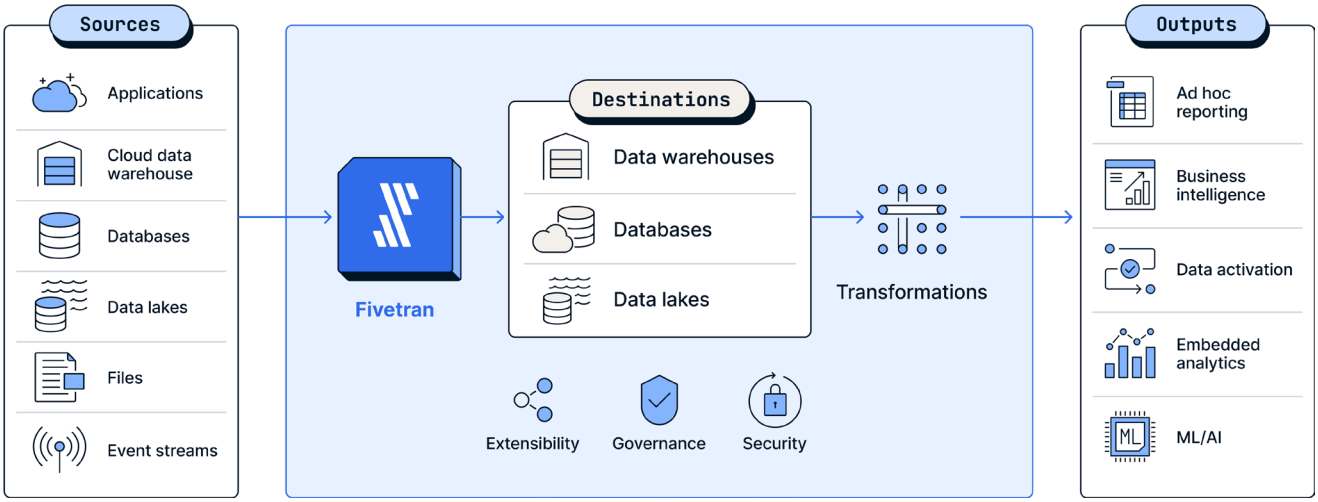
By contrast, generative AI (often called GenAI) concerns models trained on massive data sets that can mimic human reasoning. GenAI differs from other forms of machine learning and artificial intelligence in its capacity to create — i.e. generate — new data in all forms of media, including text, code, images, audio, video, and more. GenAI is a powerful multiplier for all creative or intellectual business activities, decreasing the time and effort required to manipulate information in every way — summarizing, ideating, and iterating.

Agentic AI combines the reasoning capabilities of GenAI with the ability to communicate with or operate other systems, resulting in the ability to autonomously plan and execute complex workflows that require judgment or problem-solving ability. It is fundamentally a tool for business process automation. While GenAI is a powerful productivity aid, agentic AI can be more like a virtual employee.

Architectural differences

One basic difference between traditional, generative, and agentic AI lies in the required data architecture.

The architecture for traditional AI consists of the following:



First, data is produced in various sources, such as applications, operational databases, event streams, and files. Then, an automated data pipeline extracts and loads it into a destination such as a data lake or data warehouse. Data lakes have historically been favored for AI because they offer scalability and support for unstructured data. From there, the data is transformed into formats that are legible to downstream models.

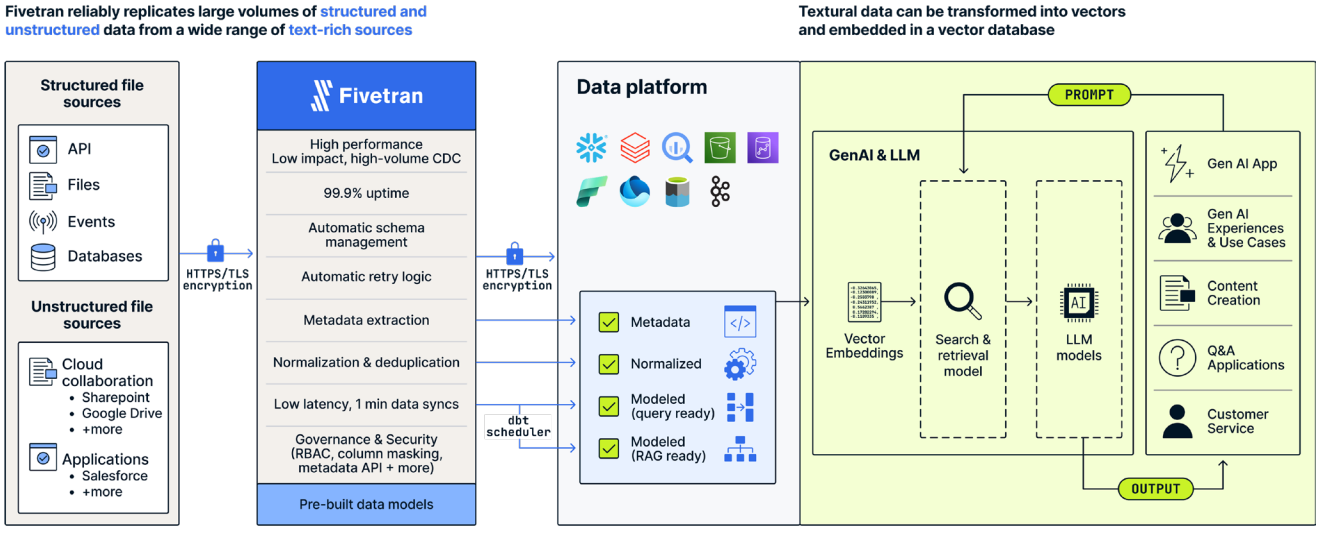
GenAI, however, involves an additional series of steps. GenAI projects commonly leverage an architecture called retrieval-augmented generation (RAG), in which a foundation model — an existing model trained on a large corpus of (usually) public data — is augmented with additional context stored in a vector database and/or knowledge graph, specialized repositories that make data intelligible to a foundation model.

Why unstructured data matters

Much of an organization's essential knowledge is not captured through transactions recorded in tables but in text-rich media such as documentation, presentations, blog posts, emails, chat logs, and correspondence.

In some fields, media — such as images, video, audio, 3D models, and other digital assets — make up important foundational knowledge or intellectual property. This includes use cases like computer-assisted design, drug discovery, and media production.

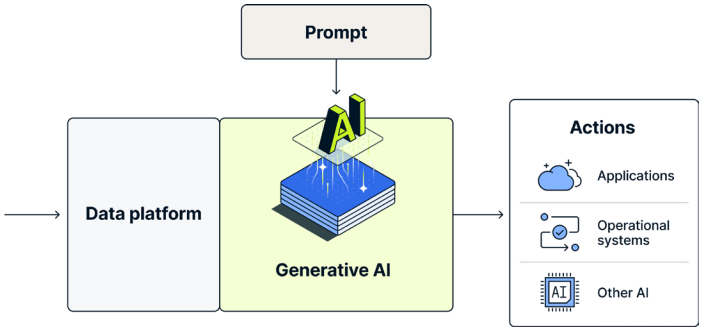
AI can be used to not only parse and analyze unstructured data (traditional AI) but also for ideation (GenAI) and as the basis for autonomous decisions (agentic AI).



Additional steps for GenAI

1. In the destination, the data is further transformed in preparation for embedding or vectorization — converted into high-dimensional numerical representations that can be stored in a vector database and compared by similarity. This may involve consolidating text-rich fields.
2. Some tools bundled with major destinations, such as Databricks Mosaic AI, Snowflake Cortex, and Google Vertex AI, abstract away the following steps. Otherwise, you must build a retrieval model, separately provision the backend for your retrieval model, and embed and load the data yourself.
    a. A retrieval model is middleware that ensures your foundation model can communicate with your vector database and/or knowledge graphs.
    b. The most common option is to turn data into embeddings or vectors and load it to a vector database. Data can also be converted to the RDF (resource description framework) format and loaded into knowledge graphs. Knowledge graphs ensure deterministic results by encoding facts, but may be less performant and more computationally intensive than vector databases due to the complexity of graph traversal.
3. Users will need an interface to interact with the product. You can set up a web application or embed it in an existing application through an API.

The architecture of agentic AI builds on the same foundation laid by GenAI, adding the ability to act through integrations with applications, operational systems, and other AI models and agents.



These systems may include commercial web applications, SaaS, programmatically controlled hardware, and even entire networks of AI agents performing specialist roles.

Data centralization is the key to supporting AI

To support analytics and automation, AI must accurately reflect ongoing developments, requiring large volumes of high-quality, up-to-date data. The traditional approach to data integration consists of extracting, transforming, and loading data (ETL), often accompanied by on-premises or self-managed data infrastructure and pipelines. Self-managed ETL poses several difficulties: scalability, guaranteeing high levels of performance and reliability, and adapting data pipelines to changing data sources and emerging needs. These difficulties limit the scale and reliability of self-managed ETL data pipelines.

The key to enabling AI is automated data integration with an easy-to-use data infrastructure. This requires rethinking the architecture of data integration — specifically, using extraction, loading, and transformation (ELT) instead of ETL. ELT requires no assumptions about how data will be used once it arrives at its destination, leading not only to more fault-resistant pipelines but also enabling data teams to centralize the data in a repository before determining what to do with it. This enables teams to more readily prototype new projects, including AI.

A central data repository should be able to accommodate both operational and analytical use cases by supporting all forms of data — structured, semi-structured, and unstructured — and through interoperability with a wide range of tools and platforms.

Data lakes — once catch-all repositories with a tendency to become unnavigable, ungoverned "data swamps" — have become increasingly capable with support for open table formats, enabling the use of structured data and technical catalog integrations, making it far easier to maintain a comprehensive inventory of a data lake's contents. Examples of modern offerings include Amazon S3, Azure Data Lake Storage, Microsoft OneLake, and Google's Cloud Storage.

The upshot is that any data architecture for AI depends on a foundation of automated data integration with the following capabilities:

- **Automation** – Data integration should require minimal human intervention. Instead of building and maintaining data pipelines, data teams should address last-mile AI problems, like ensuring high-quality outputs.
- **High performance and reliability** – Data provided to a model should be fresh and up-to-date to ensure that responses reflect ongoing developments.
- **Support for a wide range of sources** – Effective analytics requires combining data from a wide range of sources to improve the accuracy of model outputs.
- **Extensibility** – The scope and scale of AI require programmatic control, compatibility with new technologies, and the ability to expand data integration to new sources.

- **Deployment flexibility** – Many organizations maintain on-premises data infrastructure for regulatory and performance reasons. The best data integration tools offer multiple deployment options, combining automation with compliance.

Don't reinvent the wheel

With the growing importance of AI, data professionals should focus their efforts on implementing practical data products. Automation, through off-the-shelf tools and technologies that address the solved problems of data integration, offers the most practical way to rapidly bring a data team to the last mile of testing and building AI.

According to Wakefield Research, data engineers spend about 44% of their time building and maintaining data pipelines, drawing attention away from higher-value projects. Dimensional Research echoes these findings, with 44% of data professionals reporting that key data remains inaccessible and 68% of respondents reporting that, with more time, they could extract more value from their data.

The argument to buy, rather than build, data integration ultimately boils down to the following considerations:

1. **Time and effort spent building** – Building a data pipeline is a deceptively complex engineering effort, involving a granular understanding of a data source's underlying data model as well as technical considerations such as incremental updates, idempotence, buffering, scaling, parallelization, and more.
2. **Time and effort spent on maintenance** – Data pipelines often require extensive reworking when upstream data sources or downstream data needs change, creating bottlenecks to integrate new data sources.
3. **Opportunity costs** – Engineering hours spent on data pipeline building and maintenance cannot be spent on other high-value and innovative AI projects.
4. **Sustainability and scalability** – As organizations adopt a growing roster of data sources, the engineering effort required to build and maintain bespoke data pipelines becomes increasingly untenable.

Your first AI project

Before you start an AI project, make sure the solution matches the problem. Challenges like forecasting may be more easily solved using heuristics like rolling averages than by predictive modeling. Simple analytics don't call for GenAI. For simple workflows, rule-based automation is far simpler and easier than agentic AI.

Instead, identify problems that are best solved using the unique capabilities of each type of AI.

Traditional AI

Traditional AI broadly falls into 3 categories:

- **Unsupervised learning** does not require labeled data with known inputs and outputs but instead uncovers previously unknown patterns. It is most useful for open-ended pattern recognition, like clustering similar observations, detecting anomalies, or identifying correlations.
- **Supervised learning** requires labeled data showing known outputs with known inputs. The goal is to construct a model that maps inputs with outputs to predict (extrapolate or interpolate) future outputs. Linear regression is the most basic example of supervised learning. Supervised learning also includes classification use cases.
- **Reinforcement learning** incentivizes artificial agents to engage in reward-maximizing behavior through trial and error. This is used for applications that require automatic, on-the-fly decision-making, such as self-driving vehicles, gaming bots, and elevator scheduling.

Your first non-GenAI use case should concern a well-understood domain, be simple and tractable, and have high predictive power. Financial forecasting is a great example because the data sources are likely to be among the first your organization integrates and can be analyzed using straightforward methods such as linear regression. You likely already forecast things like burn rate, revenue, profit, etc. You can corroborate the results of your predictive modeling with financial modeling heuristics like percentage of sales, moving averages, and the Delphi method.

The tools and technologies required for such an analysis are also modest. They largely consist of the data stack mentioned above — financial data sources, a data lake or data warehouse, and a transformation tool — and an interactive web-based computing platform, aka "notebook" for a common language like Python or R. Notebooks allow stakeholders to visualize and collaborate on data science projects and are often bundled with major cloud data platforms like Databricks, Snowflake, Microsoft, and Google Cloud.

More complicated use cases may require multiple methods. At Fivetran, one of our internal ML use cases uses natural language processing to extract and tokenize keywords from error messages. Then, we use k-means clustering to group error messages into types, which we then label, attribute to recorded failures, and prioritize based on estimated customer impact.

Examples

Freight forwarding company Sennder gains reliable access to data to train ML models to predict costs for shippers and personalize offers to carriers.

Interloop uses Fivetran Managed Data Lake Service to scale and accelerate analytics and ML for clients.

Generative AI

GenAI can create valuable, innovative solutions for both internal and external use cases:

- Automatically producing financial documentation
- Enabling retail customers to virtually "try on" clothing and accessories
- Accelerating drug discovery by uncovering new molecular structures
- Accelerating legal and paralegal work, such as legal research
- Enabling doctors to devise personalized treatment plans
- Helping engineers optimize hardware geometry through generative design
- Generating assets for video games, films, and other digital media
- Helping educators personalize lesson plans

However, you should start with a simpler initial GenAI proof of concept with the following characteristics:

- An internal use case, limiting brand risk and exposure of sensitive data to unauthorized parties
- A simple but concrete problem that can lead to measurable productivity gains
- Augmenting a large language model (LLM) with your company's existing text-rich sources

We recommend that you consider tasks in your company's workflow that involve some combination of information retrieval and simple judgment (and some level of human annoyance).

Fivetran's first GenAI project, an internal chatbot called FivetranChat, met all these criteria. FivetranChat has not only led to very high user satisfaction and productivity but, more importantly, a low-cost alternative to manually sifting through documents or asking for another person's time — $0.10 per query.

Cloud data platforms offer tools like Databricks Mosaic AI, Snowflake Cortex, and Google Vertex AI that abstract away the need to provision a specialized database for your retrieval model or create and load embeddings. All that's left is to write middleware connecting your AI tool with a front end. Otherwise, you will additionally need to extract, embed, and load data from your data lake to a vector database or graph database you have separately provisioned.

Even so, the bulk of the technical effort for your RAG project will go toward testing and validating results.

Examples

CRM platform HubSpot uses Fivetran to centralize employee performance data, including text-based information, enabling the analysis of employee performance and improvement of management practices.

National Australia Bank uses real-time data integration by Fivetran to enable GenAI projects like a banker chat assistant.

Agentic AI

An AI agent is given a goal, and then plans and executes a solution in response to it. AI agents may write to applications and databases, search the web, write and execute code, and even operate physical systems.

Architecturally, agentic AI is an extension of GenAI and will require many of the same tools and platforms. Your team may need to build a considerable amount of middleware (i.e. API integrations) to give the agent access to the necessary tools.

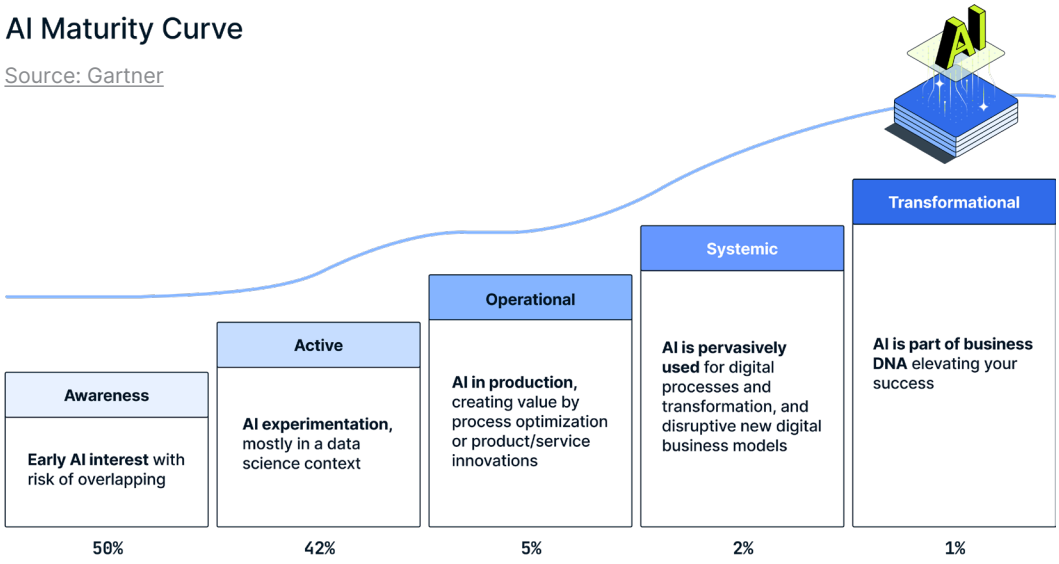
The sophistication of an AI agent ranges from simple workflow automation — in which a system performs a fixed, linear set of steps — to fully autonomous goal-oriented behavior, which might require repeated iterations and decisions to fulfill an objective.

Your first agentic AI project should augment a workflow that requires human-like discretion and judgment but is bound enough to be easily tested, can be easily reversed, and doesn't endanger anything sensitive. One such use case is to transcribe meeting notes and then automatically ticket, prioritize, schedule, track, and update projects.

Agentic AI is complex and expensive. Make sure you have exhausted the capabilities of traditional AI and GenAI (not to mention automation based on rules and heuristics) before you consider agentic AI.

Data readiness for AI

None of these AI projects will be possible without a solid foundation that yields AI-ready data.



Data centralization is the key prerequisite to AI readiness, enabling teams to prototype and deploy analytics projects of all kinds. Accuracy and business value increase with the volume, variety, and relevance of data. To fully ascend the AI maturity curve, your data must be:

- **Clean** – Data must be de-duplicated, follow standardized naming conventions and formats, and be free of errors. You will also need a process for identifying and dealing with missing and outlier values.
- **Compliant** – Ethics and regulatory compliance are crucial. Ensure data is AI-ready by blocking or hashing sensitive information and managing access control.
- **Fresh** – Data must be fresh and up-to-date to ensure that responses are representative, accurate, and relevant.

An additional consideration is the difference between structured, unstructured, and semi-structured data.

Traditional AI, like predictive modeling, is typically performed on structured data — fields organized into tables or markup documents. Structured data usually records transactions performed by applications and database backends.

However, most of an organization's data is unstructured, consisting of text, images, code, video, audio, and other digital assets found in correspondence, documentation, knowledge bases, marketing collateral, code bases, asset libraries, and other sources. This unstructured data contains a wealth of insight, much of it qualitative and difficult to capture in a table. Generative and agentic AI depend heavily on unstructured data for reasoning and pattern recognition. LLMs, for example, are trained extensively on large bodies of text.

Some data is semi-structured — a table or markup document may, for instance, contain large text fields. Semi-structured data can be readily converted into unstructured data as needed.

Bringing your AI project to production

The old statistics cliche of “garbage in, garbage out” applies to all AI. The cleverest algorithm can’t overcome the problems that arise from bad data — data that is duplicated, inaccurate, irrelevant, dated, or otherwise distorts a model, producing not only misleading outputs but real business losses.

Data curation

You must carefully select and refine the data used for AI to ensure that your model is provided with relevant, accurate context and not noise. A (by no means comprehensive) list of best practices:

- Ensure that the data comes from credible, authoritative sources. More data is not always better, especially if the data is misleading, non-representative, factually incorrect, outdated, or irrelevant.
- You must also anonymize, encrypt, or exclude sensitive data.
- Ideally, the data is tagged with metadata, offering additional context.

QA and tuning

Any AI project you deploy should allow users to provide feedback so that you can identify your model's shortcomings. GenAI, for instance, may hallucinate, producing plausible but misleading outputs.

Three criteria for evaluating the quality of an AI model include:

- **Response quality** – Does the model adhere to the set of expected outputs and produce coherent results? For GenAI models, does it demonstrate the ability to accurately synthesize the information it has been provided?
- **Latency** – Does the model produce results quickly enough to meet the demands of a particular use case? Can it handle simultaneous requests?
- **Error handling** – Does the model produce incomplete outputs, format results inappropriately, or fail to respond altogether? How does the model communicate failure states?

Other considerations for improving AI outputs include hyperparameter tuning, algorithm selection, and the fine-tuning of foundation models — all beyond the scope of this guide.

Prompt engineering

Prompt engineering is a unique consideration for GenAI. Prompt design and prompt engineering concern crafting inputs — prompts — that elicit helpful responses and systematizing and governing the use of prompts, respectively.

Good prompts minimize ambiguity and provide as much useful context as possible in the following ways:

1. **Give directions** – Add adjectives, context, and other descriptors and guidance about your intended output. This includes adding the user's persona in the prompt — you can tell the model what to emphasize and what detail to offer. Without explicit examples, guidance, or broader context, a model is likely to produce unhelpful results.
2. **Provide examples** – The quality and consistency of responses can be vastly improved by providing both positive and negative examples for the model.
3. **Format the response** – Explicitly describe the format of the response. If you need information presented as a numbered list or bullet points, say so explicitly. This consideration is especially important for generating code or data structures like JSON.
4. **Divide labor** – Not everything has to be accomplished with a single prompt. You can take advantage of your session's short-term memory with a succession of prompts that add additional details, direction, formatting, etc. to continuously refine the output. You can import the output from one model into another, more specialized one to further refine your results.
5. **Evaluate outputs and iterate** – Prompting should be treated as an iterative process. The "engineering" side of prompt engineering is a matter of systematically testing, evaluating, and iterating through different configurations of prompts.

Sample Prompt

```
The following is an example of a
naive prompt:

Write me a script to move data from
a CSV file to a data lake.

---

A more helpful prompt includes
additional context. You may even
attach files:

You are a data engineer. Provide
Python code that extracts and
loads a schema from a set of CSV
files to Apache Iceberg tables in
a data lake while reflecting data
engineering best practices.

Import external libraries as
needed.

Use the attached CSV files for
reference.
```

You may also consider preprocessing your prompts to eliminate abusive language and other undesirable behavior.

Enforcing predictability

The RAG architecture used in GenAI commonly leverages vector-based information retrieval, which produces results probabilistically, sometimes leading to inconsistent or erroneous outputs. An alternative is knowledge graphs, which provide a more deterministic approach, encoding factual relationships to ensure accuracy. However, graph traversal is inherently computationally expensive and is not commonly a first choice for GenAI projects.

Together, vector-based information retrieval, the ability to rate and provide feedback on outputs, and practices such as constant tuning, prompt engineering, and data curation may be enough to ensure a high prompt success rate. Make sure your retrieval model allows users to submit detailed feedback, and make sure your team includes engineers with the technical expertise to fix mishaps. Human supervision in the loop may be necessary to test the answers provided by a model until it meets an acceptable threshold for accuracy and relevance before an application is deployed.

AI continues to become more accessible

AI remains a live area of research in both academic and corporate labs, and there is constant architectural innovation in modeling, information retrieval methods, and the embedding of AI systems. Well-established use cases such as support chatbots will help you lay the groundwork for riskier and more demanding AI pursuits by providing valuable experience with the requisite technologies and last-mile refinement required to bring an AI project to fruition. For all the technical complexity of AI, it fundamentally begins with a solid foundation of data centralization. The steps outlined below should be sufficient to start your first AI project.

Get started with your foundation for AI success

Fivetran makes it easy to build a foundation for AI success by extracting, loading, and transforming data with no coding or engineering effort.

1. The first step is to sign up at https://fivetran.com/signup to get a 14-day free trial, effective after your initial data sync.
2. Then, navigate the interface to choose connector sources as well as destinations. In both cases, you will supply credentials to a setup form to connect.
3. Set a schedule and begin your initial sync.
4. Use the relevant Fivetran Transformations to prepare the data for AI. The Unified RAG dbt Package extracts and combines text-rich fields to turn into embeddings.

For the numerical and categorical prediction characteristic of most traditional AI, transformations will primarily take on the form of feature engineering — identifying variables that are likely to have predictive value, finding variables that are correlated with each other in order to combine or reduce them (i.e. feature reduction), and so on. For supervised learning, you will produce tables that combine features with targets or output/dependent variables.

For generative and agentic AI involving LLMs, you will need to transform data into a format legible to a vector database or knowledge graph. The first step is to combine text-rich fields together using SQL, which is readily achieved through the Unified RAG dbt Package.

Fivetran

Fivetran, the global leader in data movement, helps customers use their data to power everything from AI applications and ML models, to predictive analytics and operational workloads. The Fivetran platform reliably and securely centralizes data from hundreds of SaaS applications and databases into any cloud destination — whether deployed on-premises, in the cloud or in a hybrid environment. Thousands of global brands, including Autodesk, Condé Nast, JetBlue and Morgan Stanley, trust Fivetran to move their most valuable data assets to fuel analytics, drive operational efficiencies and power innovation.

For more info, visit Fivetran.com.