Best Practices for Building Successful AI Solutions

29 April 2024 - ID G00812933 - 19 min read

By Birgi Tamersoy

Initiatives: Analytics and Artificial Intelligence for Technical Professionals

Artificial intelligence has immense potential to improve the lives of people around the world. A successful AI needs to be accurate, robust and reliable. Data and analytics technical professionals should follow the best practices outlined in this analysis to build successful AI solutions.

Overview

Key Findings

- Building a successful AI solution entails more than training an AI model. The engineering complexity and effort associated with deploying robust and reliable AI solutions is often underestimated.
- Starting an AI development process without thoroughly exploring the data is a common mistake, as is the assumption that training data or pilot environments reflect real-world scenarios.
- All AI models try to learn the things that remain the same in each problem (the “invariances”). Spurious variations in the data or increased problem space makes learning both more difficult and data hungry.
- In production, how an AI system fails can be as important as how it succeeds. When an AI system blunders, the users lose trust in the system and regaining that trust is difficult.

Recommendations

- Have an “AI system view” when developing AI products. Identify potential shortcomings of your AI model in terms of accuracy, robustness, and dependability, and introduce complementary system components that address these shortcomings.
- Start the AI development process by thoroughly exploring the data, not by AI model training. Know and document the assumptions you make during the pilot phase and also document the restrictions associated with these assumptions.
- Minimize your “ask” from each AI model by identifying the minimum task coverage it needs to have and train accordingly. Eliminate any spurious variations in the data during the preprocessing step.
- Ensure “graceful failure” by using “composite AI” approaches. Accompany your complex AI models with simpler and more interpretable AI models to reduce and control the errors. For high-risk applications, introduce human-in-the-loop controls.

Analysis

AI has immense potential for improving the lives of people around the world. Its transformative power comes from the ability to process large amounts of data efficiently and effectively. It can address many of the major global challenges, such as the resource/demand gap in healthcare, efficiency problems in agricultural production, or the inequalities in education. Organizations in every industry can leverage AI to gain more insights in their domains, improve their efficiencies, and increase the quality of their products and services.

Gartner’s 2023 AI in the Enterprise Survey showed that only 9% of organizations identified as “mature AI organizations.”[1] This is where AI is used in multiple business units and processes of the organization, more than five AI use cases are deployed, and AI use cases stay in production for more than two years. 74% of the respondents stated that they were planning to deploy AI solutions within two years.

As they move from experimenting with AI pilots to deployment of AI solutions, organizations face a number of barriers, including:

- Technical difficulties with implementation and deployment.
- Concerns about the reliability and robustness of the deployed solutions.
- Availability of talent and related technological skills.

All AI projects go through five major stages in their life cycle (see Figure 1):

1. Selection and design
2. Development

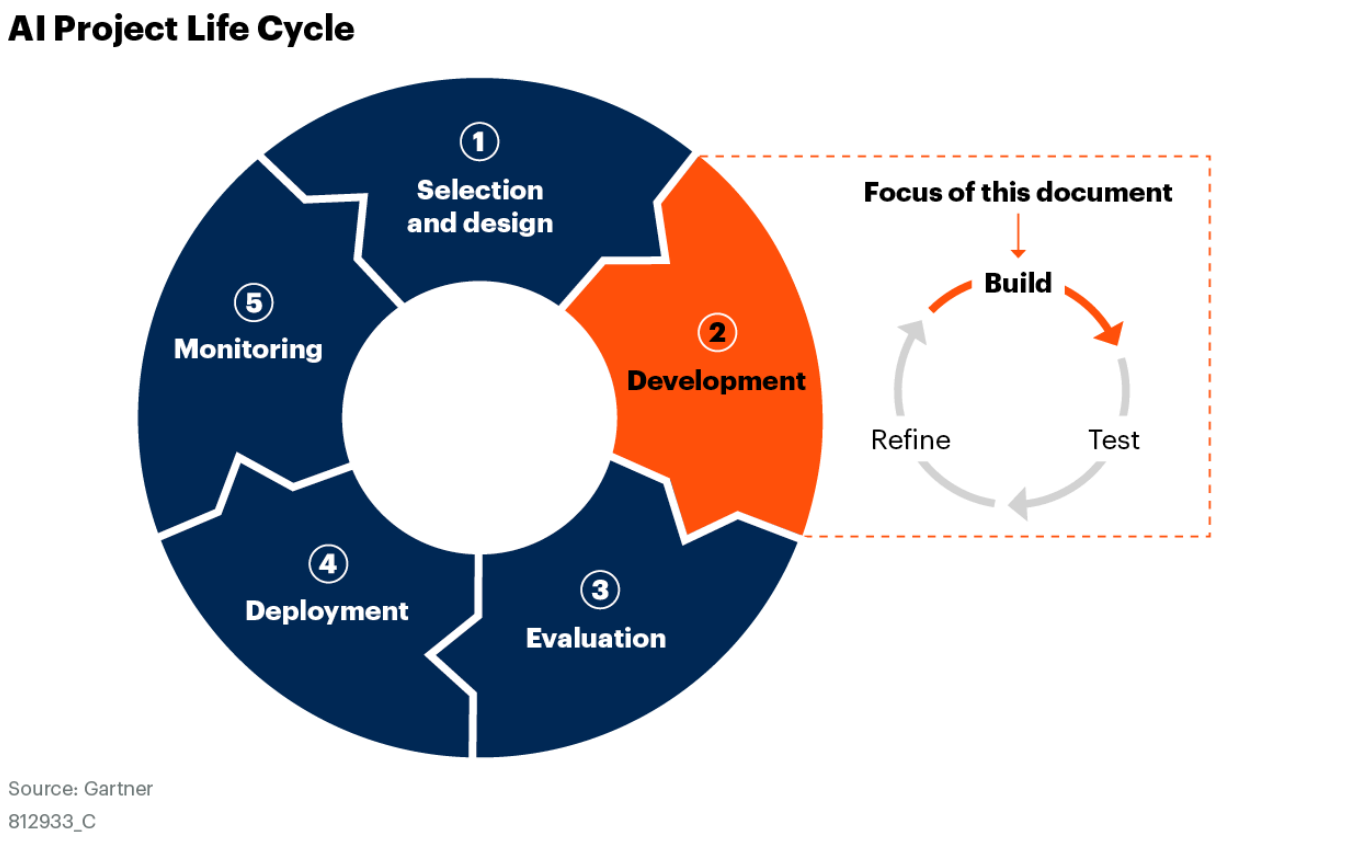
3. Evaluation

4. Deployment

5. Monitoring

The development stage is an iterative process, where data scientists, machine learning (ML) engineers and AI architects build, test and refine solutions in multiple iterations. This analysis focuses on the "build" phase of the "development" stage, and provides best practices to technical professionals for building accurate, robust and dependable AI solutions.

Figure 1. AI Project Life Cycle



Best practices covered in this analysis are presented in Figure 2:

1. Know your problem, user, environment and data.

2. Have an AI system view, instead of an AI model view.

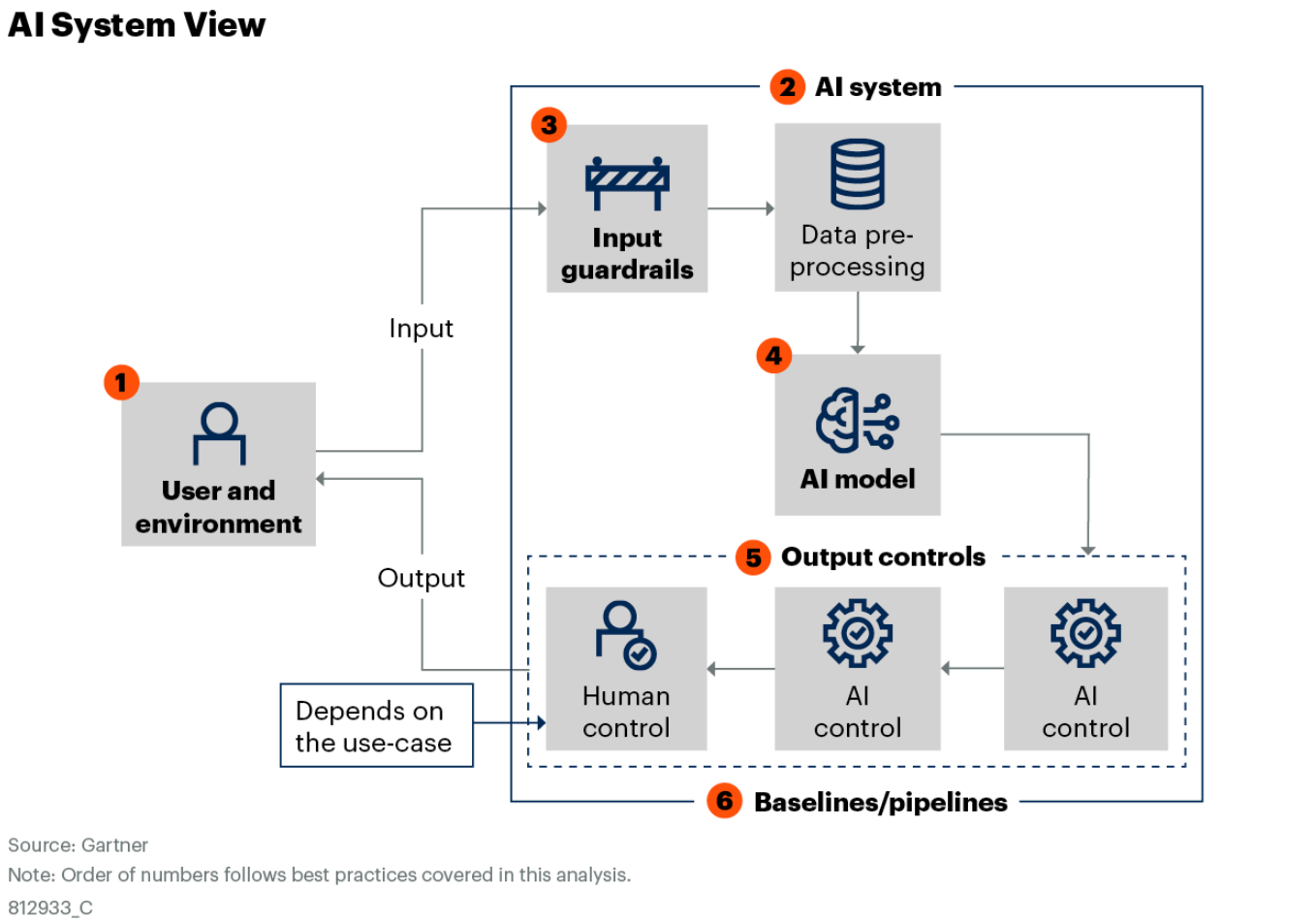
3. Limit what goes into your system using explicit input guardrails.

4. Minimize what you ask from your AI models.

5. Ensure graceful failure through explicit output controls.

6. Get your baselines and pipelines working first, and focus on increasing complexity later.

Figure 2. AI System View



Source: Gartner
Note: Order of numbers follows best practices covered in this analysis.
812933_C

Gartner

Know Your Assumptions and Thoroughly Analyze Your Data

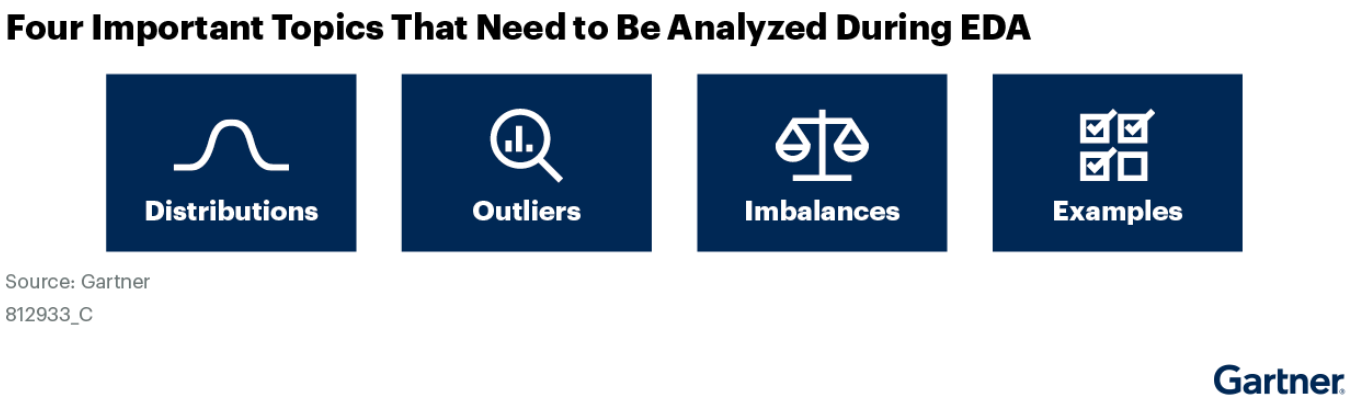
When developing AI solutions, technical professionals tend to make a number of assumptions about domain boundaries, target environments, user behavior and data distributions. Even the selection of a particular ML algorithm (for example, a decision forest versus a neural network) entails making assumptions. Knowing which assumptions are being made and understanding the consequent limitations that will arise in production are two of the most important skills a data scientist, an ML engineer or an AI architect should have.

Knowing your assumptions is especially important as you move from AI pilots to AI products. Imagine a hypothetical scenario where an automatic office-cleaning robot is being piloted. The robot is fitted with high-resolution color cameras and equipped with accurate AI-based navigation and control systems. The robot performs well in the laboratory experiments during the pilot phase.

After the product release, customers deploy the robots and experience a complete failure because customers prefer the cleaning to be done during the night when the office lights are turned off.

Once you understand the problem thoroughly, you should work to understand the data thoroughly. One useful technique is exploratory data analysis. This includes four important topics highlighted in Figure 3.

Figure 3. Four Important Topics for Exploratory Data Analysis



- **Analyze the distributions:** Summary statistics (like mean and median) do not provide the full picture and may lead to incorrect insights. Analyzing the complete data distributions will provide better insights and drive the decisions on what assumptions can or should be made in the subsequent development phases. A simple example would be a laboratory test which returns +1 in presence of a specified property and –1 in its absence. Assuming there are an equal number of samples with and without that property in a dataset, the mean will be 0. Looking at the mean alone will not provide the insight that the dataset has two peaks — one at +1 and the other at –1.

- **Analyze the outliers:** When outliers in the data are not handled appropriately, many AI solutions can be thrown off, both during training and in deployment. Investigating the outliers, knowing their sources, and being diligent about how to manage them (whether to group them, remove them or ignore them, for example) will help with designing system components that will increase the robustness of the overall system.

- **Analyze the imbalances:** A *very* unbalanced dataset requires a different set of techniques in processing compared to a *moderately* unbalanced dataset. For example, in a dataset of credit card transactions, the number of fraudulent transactions will be significantly less than the number of legitimate transactions, whereas the number of transactions in major spending categories (shopping, dining or traveling, for example) will be relatively less unbalanced. Establishing the level of imbalance, and introducing the appropriate techniques (like over/under-sampling) and metrics (like weighted losses) will reduce unexpected results in deployment.

- **Analyze the examples:** Examining the trends, patterns, input and output ranges, and the variability in the data, will help with building insights about the problem complexity. When building successful AI solutions, it is important to match the model complexity to the problem complexity. For example, scoring the severity of tumors in mammograms is a difficult image-processing problem and it would require a complex model to learn this input/output relationships. Whereas determining the orientation of a manufactured part on a production line — under controlled lighting and with a consistent background — is a much simpler image-processing problem that would require a simpler model.

Have an "AI System View" Instead of an "AI Model View"

In successful AI solutions, the *primary* AI model should be supported by a number of auxiliary components, such as input guardrails and output controls. The high-level "AI system view" presented in Figure 2 applies to *all* AI solutions, whether the primary AI model (that is, the *brain* of the system) is a modest traditional ML model or a colossal generative large language model (LLM). The nature and the quantity of the auxiliary components depend on the capabilities of the primary AI model, and the robustness and reliability requirements of the use case.

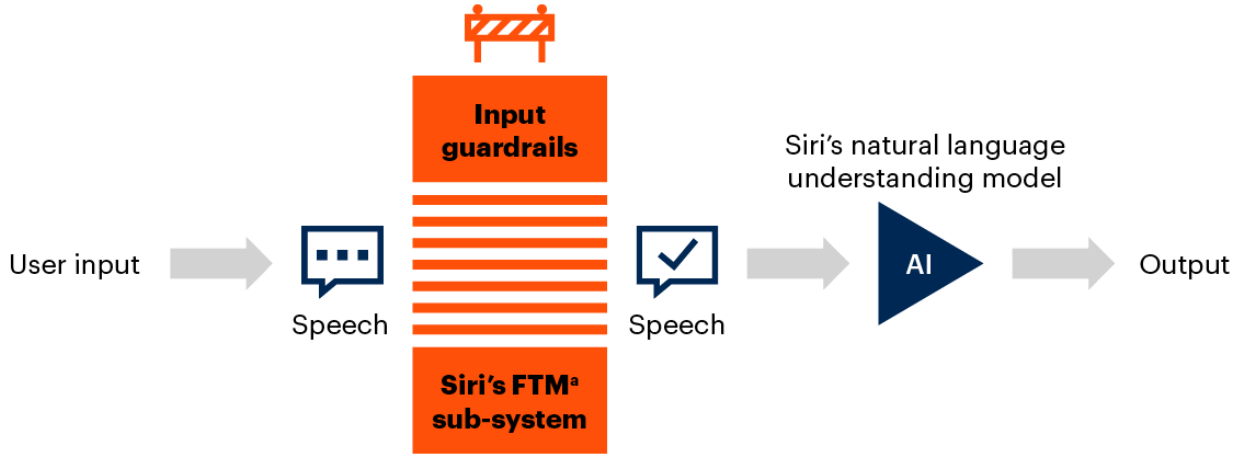
Limit What Goes Into Your System Using Guardrails

When AI models are exposed to inputs that they have not seen before, their outputs tend to be less predictable. Input guardrails improve the robustness of an AI model by eliminating (or in some cases by modifying) potentially problematic inputs.

Input guardrails can be as simple as rule-based range checks, or as elaborate as composite AI subsystems. A good example of the latter is Apple's digital assistant. Siri has a False Trigger Mitigation (FTM) subsystem that acts as a barrier between the user input and Siri's natural language understanding model (in other words, the primary AI model) [2] (see Figure 4).

Figure 4. The Use of Input Guardrails to Eliminate Problematic Inputs

The Use of Input Guardrails to Eliminate Problematic Inputs
Illustrative



Source: Gartner
[a] False trigger mitigation
812933_C

Gartner

The FTM subsystem is tasked with filtering out speech and background noise when listening for trigger words. For example, when the user says "Are you serious about that?" in a conversation, the FTM subsystem tries to avoid the confusion between "serious" and "Siri," and avoids a false trigger of the system.

The FTM subsystem consists of three separate AI components:

1. **Lattice RNN:** A recurrent neural network (RNN) applied on the automatic speech recognition (ASR) decoding lattices.
2. **Acoustic false trigger mitigator:** A streaming transformer encoder applied to acoustic-only features.
3. **Out-of-domain language detector:** A transformer-based natural language understanding model modified into a classifier.

Any user input goes through these three separate checks and the results are then incorporated in another "neural combiner" to determine whether the input should be passed to the primary AI model or not. As the FTM example demonstrates, the complexity and the effort required for developing input guardrails should not be underestimated when building robust AI solutions.

Minimize the "Ask" From Your AI Models

AI models try to learn the "invariances" in each problem. For example, when training an AI model to detect the traffic signs in color images, the goal is to learn the invariant visual properties of traffic

signs. In other words, how traffic signs look in different lighting conditions, different weather conditions and from different viewpoints.

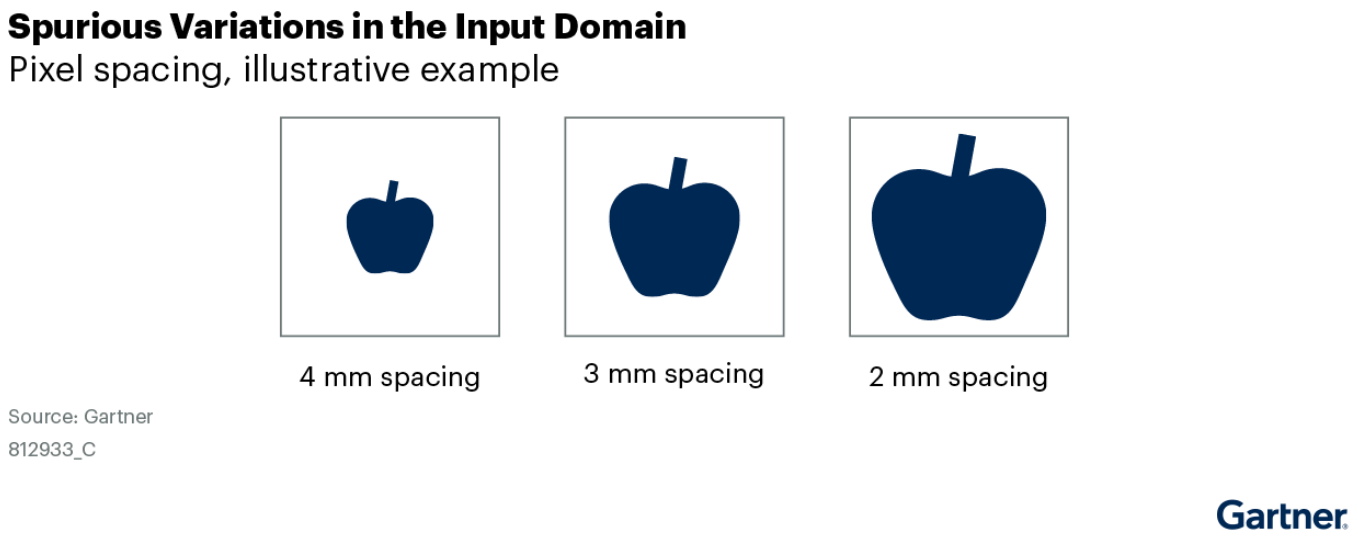
Variations in the input domain can be spurious (like two cameras capturing images in different resolutions) or can be inherent in the problem (when weather conditions change how traffic signs look in images). When training robust AI models, all variations in the input domain should be covered in the training dataset. For the traffic sign detection problem, this would mean providing images of traffic signs in all lighting conditions, all weather conditions, and from all viewpoints.

When training robust AI models, having diverse, balanced and representative datasets is important. The required quantity of data can be reduced by limiting any variations in the input domain. Spurious variations can be weeded out during data preprocessing, whereas problem-inherent variations can be reduced by limiting the scope of the AI models. For example, a model designed to detect only the stop signs will require less training data than a model that is designed to detect all traffic signs, traffic lights and moving objects together.

Reduce Spurious Variations in Data Preprocessing

To reduce spurious variances in the input domain, you can use normalizations, standardizations or more elaborate data transformations. Figure 5 illustrates a common example from medical imaging.

Figure 5. Spurious Variations in the Input Domain



Pixel spacing is an important scan parameter that represents the actual distance covered by a pixel of an acquired image. In the three cases shown in Figure 5, the same apple is imaged, but with different pixel spacings. As the pixel spacing does not have any correlation with the actual phenomena (the anatomy of the apple), it is a spurious variation and can be eliminated during preprocessing by resampling all inputs into a common pixel spacing. Or in other words, by making

all the images look the same size. This allows the model to focus on more important image features during training.

Limit the Scope of AI Models by Task Specialization

Addressing multiple tasks at the same time is a complex problem for an AI model to learn. For example, a single AI model might be tasked with determining if a text input has positive or negative sentiment, summarizing the input, and translating it into another language. That AI model needs to be trained with a dataset that covers all of the variations in all of these problems. Task specialization reduces the problem complexity by limiting the inherent variations that need to be considered during development. A simpler problem with less inherent variances can be solved by a simpler AI model using less training data. In production, simpler models are easier to maintain, more efficient to serve, and provide more robustness (as both input and output can be controlled more effectively).

A good example of task specialization is the vision-based Tesla Autopilot system. [3,4] Even though inputs from several cameras are incorporated into a common implicit representation of the environment, individual tasks (such as detection of pedestrians, detection of road signs and detection of moving objects) are done using separate AI models. The complete system consists of 48 neural networks.

It is important to note that generalist AI models — such as foundation models in natural language processing (NLP) and computer vision (CV) — have tremendous benefits. LLMs in NLP provide generic natural language understanding and generation capabilities. Foundation models in CV provide valuable feature extractors. Foundation models also offer an exceptionally good starting point for training specialized AI models, but they are more difficult to control, and significantly less efficient to serve in production.

Ensure Graceful Failure Through Composite AI

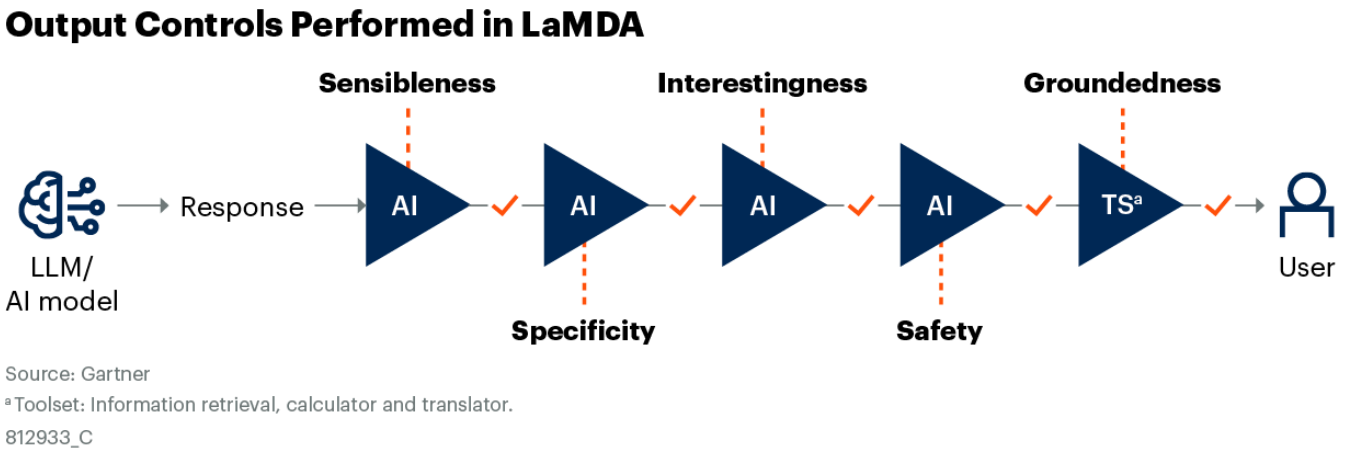
When AI systems are productized, how they fail becomes as important as how they succeed. If an AI system "blunders," users lose trust in that system — and regaining trust is difficult. Any AI system that is not adopted or trusted by the users is a failed AI system.

Data scientists, ML engineers and AI architects should focus on ensuring "graceful failure" when moving from AI pilots to AI products. This can be achieved by introducing additional components that control the output of the primary AI model. The number and the nature of these controls depend on both the sensitivity of the use case and the capabilities of the primary AI model.

Similar to the input guardrails discussed before, output controls can also be as simple as rules-based range checks, or as complex as a cascade of elaborate AI models. A good example for the latter is Google's LaMDA (the acronym stands for "Language Models for Dialog Applications"). [5] In LaMDA, the output of the LLM (the primary AI model in this case) goes through a cascade of

checks before being shared with the user. Separate and discriminative AI models are used to check whether the output is sensible, specific, interesting, safe and grounded (see Figure 6).

Figure 6. Output Controls Performed in LaMDA



For sensitive use cases (such as the ones in healthcare, judiciary and finance), additional human-in-the-loop controls may also be necessary. An example for this is when a treatment option is being decided for a cancer patient. An AI system may provide both qualitative and quantitative analysis on the patient's medical scans, laboratory tests, and longitudinal disease progression, but the final decision will be made by the group of experts in the tumor board.

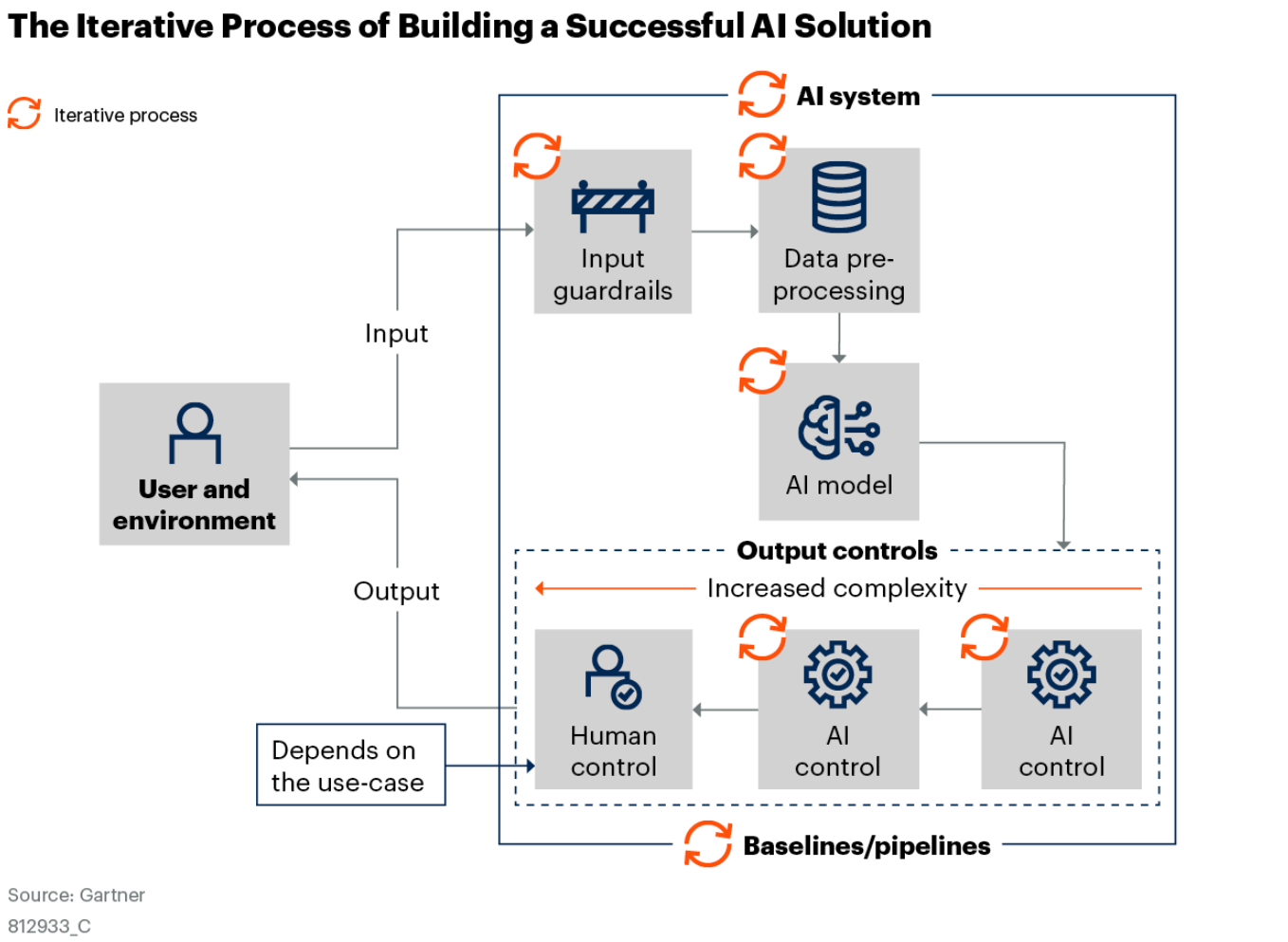
Get Your Baselines and Pipelines First, Increase Complexity Later

A successful AI solution consists of a number of components, including the primary AI model, the input guardrails and the output controls. Technical professionals should not underestimate the engineering effort and complexity of making these components work in harmony. Making sure that the data flows through all the components in an expected way should be done earlier in the development process, so that debugging and solving potential problems are easier.

A good approach is to focus on developing the baselines and building the pipelines first, and then increasing complexity in the appropriate parts of the system as needed. At any stage of the development, it is better to have a not perfect but *working* AI solution, rather than a technically state-of-the-art but *not working* AI solution.

Building a successful AI solution is an iterative process where each step brings in more insights about the problem, the user and the environment (see Figure 7). Referring back to the cleaning robot example, a new insight after the first iteration may be that the common areas, such as kitchens, need more cleaning than the small meeting rooms. Or another new insight may be that mapping rooms with a lot of chairs are technically harder than mapping rooms with a lot of sofas.

Figure 7. The Iterative Process of Building a Successful AI Solution



Increasing the complexity iteratively reduces the potential for technical debt. Again, referring back to the cleaning robot example, in the first iterations of the solution it may be sufficient to rely on an AI model that does frame-by-frame obstacle detection. A more advanced AI model that uses the relevant temporal information (gathered from multiple consecutive video frames) may potentially improve both the robustness and the accuracy of obstacle detection, but at the same time introduces additional complexity, meaning that it should probably be deferred to a later development iteration.

With less technical debt to resolve in later stages, executing more development cycles may be possible, where each cycle brings in more insights about the problem, the user and the environment. With more insights, fewer incorrect assumptions will be made and, as a result, the AI solution will become better.

Recommendations

Technical professionals tasked with building AI solutions should focus on the robustness and dependability of the solutions as much, if not more, as their accuracy.

Some of the common mistakes that technical professionals make when building AI solutions are:

- Concentrating only on the AI model and not the AI system.
- Starting the development without a thorough understanding of the problem.
- Hoping that a single AI model will solve every task related to the problem.

To transition from successful AI pilots to successful AI products, data scientists, ML engineers and AI architects should follow these best practices:

- View the AI solution as a collection of components (the AI model, the input guardrails and the output controls, for example) working together. Address any limitations or shortcomings in the primary AI model by introducing auxiliary components (this will help with building accurate, robust and dependable AI solutions).
- Do not start the AI solution development with AI model training. A thorough problem understanding, and a thorough data understanding are prerequisites for AI model training. Technical professionals must have an in-depth understanding of the differences and expectations in the development of AI pilots and AI products.
- Limit the task scope of each AI model as much as possible, understand the requirements on the quality and quantity of training data with increased task scope, and eliminate any variance that is not inherent to the problem during data preprocessing.
- User trust is easily lost, and tremendously hard to regain. Implement necessary AI controls to ensure that when the AI system fails, the failure is as graceful as possible. It may take a single failure instance to completely rule out the use of an AI solution in an organization.

Conclusion

AI has great potential in creating value for organizations across all industries. Building successful AI solutions is more than just training AI models. It requires an “AI System View” where the primary AI model is supported by a number of auxiliary components to increase the robustness and reliability of the solution. Both the number and the complexity of these auxiliary components depend on the sensitivity of the use case. They can be as simple as rules-based checks on the metadata, or as complex as complete composite AI subsystems. The success of the AI system depends on all components working in harmony.

Evidence

[1] **2023 Gartner AI in the Enterprise Survey:** This study was conducted to understand the keys to successful AI implementations and their impact on the broader AI that has been brought by generative AI. The research was conducted online from 19 October through 21 December 2023 among 703 respondents from organizations in the U.S., Germany and the U.K. The main sample consisted of 645 out of the 703. Organizations were required to have developed or intended to deploy at least two AI initiatives within the next three years. Respondents were required to be part of the organization's corporate leadership or report to corporate leadership roles. Fifty-eight out of 703 are the business intelligence (BI) sample respondents. Organizations were required to have developed or intended to deploy at least one AI initiative within the next three years. Respondents were required to be part of the organization's corporate leadership or report to corporate leadership roles or below (senior manager and above), and to be primarily responsible for BI in their organizations. Both the main sample and the BI sample respondents were required to have a high level of involvement with at least one AI initiative. They were also required to have one of the following roles when related to AI in their organizations: determine AI business objectives, measure the value derived from AI initiatives, or manage AI initiatives development and implementation. Quotas among the main sample were established for company size and for industries to ensure a good representation across the sample. No quotas were established for the BI sample.

Disclaimer: The results of this survey do not represent global findings or the market as a whole, but reflect the sentiments of the respondents and companies surveyed.

[2] Voice Trigger System for Siri — Apple Machine Learning Research, Apple

[3] Tesla AI Day 2022, YouTube

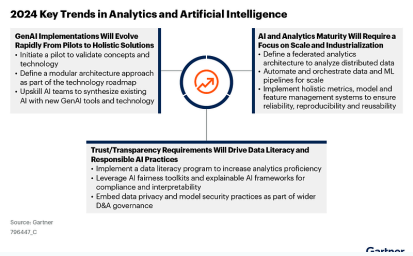
[4] AI & Robotics, Tesla

[5] LaMDA: Language Models for Dialog Applications, Google

Recommended by Author

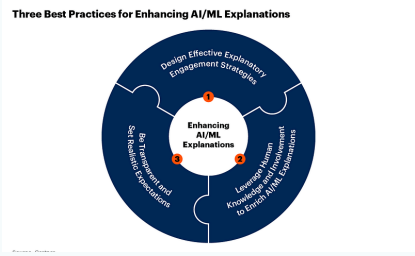
2024 Planning Guide for Analytics and Artificial Intelligence

RESEARCH ▪ 4 October 2023



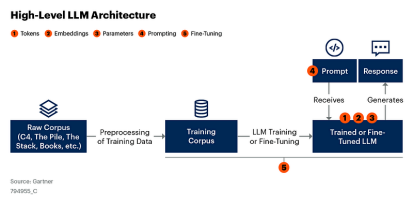
How to Explain AI and Machine Learning Models to Your Stakeholders

RESEARCH ▪ 1 December 2023



What Technical Professionals Need to Know About Large Language Models

RESEARCH ▪ 28 September 2023

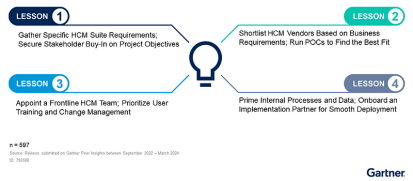


View More

Your Peers Also Viewed

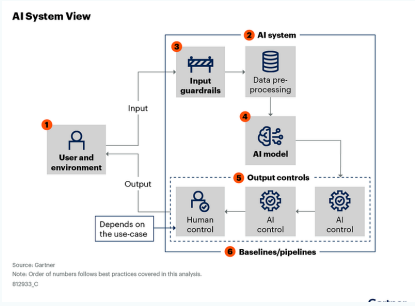
Peer Lessons Learned for Cloud HCM Suites for 1,000+ Employee Enterprises Implementation

RESEARCH ▪ 30 April 2024



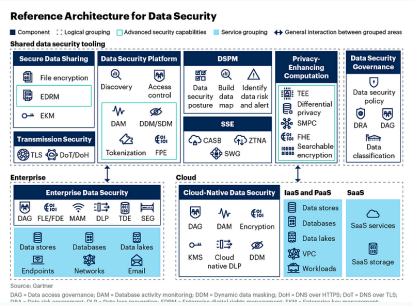
Best Practices for Building Successful AI Solutions

RESEARCH ▪ 29 April 2024



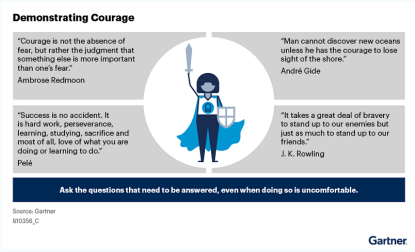
Reference Architecture Brief: Data Security

RESEARCH ▪ 26 April 2024



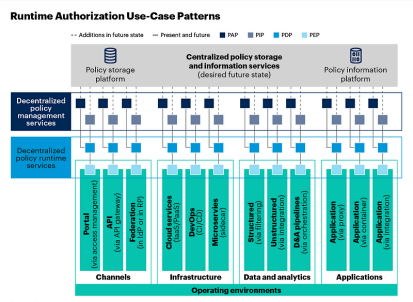
Agile Superhero Handbook

RESEARCH ▪ 26 April 2024



Architecting Modern Runtime Authorization

RESEARCH ▪ 23 April 2024



Supporting Initiatives

Analytics and Artificial Intelligence for Technical Professionals

Following