Rotational Labs, Inc

Augmenting the Firm:

A Guide to Hiring Your First AI

First Edition, March 2024

Are you a leader excited about the potential impact of AI for your business, but with no idea where to start?

If you're looking for opportunities to leverage Gen AI to solve real problems at your organization, but you're worried about the risks and the ROI, or wondering how to protect and monetize your organization's data in the age of off-the-shelf AI, this book is for you.

Executive Summary

In the era of generative AI and large language models (LLMs), the landscape of organizational strategy and operations is undergoing a profound transformation. This playbook, crafted from our practical experience implementing data products over the past decade, speaks to the dual sentiments of excitement and doubt permeating the business world. We're aiming to address the "blank canvas anxiety" many executives face, answering the daunting question of where to begin with AI.

This playbook aims to be a pragmatic guide, focused on applied solutions and paths to ROI. We've learned that technologists tend to over-index on models, often at the expense of business value. At the same time, executives care about talent, customers, processes, and products. As such, solutions marketed to executives (e.g. no-code platforms, AI chatbots, etc.) are off-the-shelf, proprietary tools that are difficult to customize and unlikely to differentiate an organization from its competitors. We challenge both technologists and executives to think beyond the obvious.

Our advice to leaders, which is essentially "**focus on the mundane"**, may surprise some readers. We are all drowning in an ocean of AI hype, and it may feel counterintuitive to be encouraged to swim ashore, but we hope this book will serve as a beacon. As we reflect on what we have observed across several dozen organizations working to adopt AI, a clear pattern has emerged: those who are willing to recognize mundane use cases are realizing the most value.

We introduce a new framework for reimagining organizations as collectives of talented people and intelligent agents, pushing the boundaries of traditional models to leverage AI for enhanced efficiency and innovation.

Welcome to a journey of transforming your organization with AI, where everyday practice meets possibility.

Editor's Note: All art in this document generated using Craiyon's AI image generator.[1]

[1] https://www.craiyon.com/

Introduction: Re-Imagining the Organization

In 1937, future Nobel prize-winning economist Ronald Coase published "The Nature of the Firm," suggesting that the value of the firm is in reducing transactional costs that would occur on the open market (constant negotiation, search, contracting) through direct managerial oversight.[2] A team of talented individuals focused on solving a problem, Coase argues, is the most effective and efficient way to find solutions.

But what, you might ask, does this have to do with Gen AI and LLMs?

As an executive, it's time to update your mental model of your team to include ***not only your talented* human *workforce, but also the* intelligent agents *who augment and support them***. Ethan Mollick, Associate Professor of Management at the Wharton School of Business at the University of Pennsylvania, calls this "co-intelligence".[3]

Intelligent Agents

By intelligent agents, we don't mean to suggest generalized "agents" that can complete complex tasks given general prompts or instructions. The hypothesis of generalized artificial intelligence presumes many assumptions about cognition and contextual awareness, and we are far from a future in which such things could be proven conclusively.

[2] Coase, Ronald (1937) "The Nature of the Firm," Economica. Volume 4, Issue 16, pages 386-405. https://onlinelibrary.wiley.com/doi/full/10.1111/j.1468-0335.1937.tb00002.x
[3] Mollick, Ethan (2024) *Co-Intelligence: Living and Working with AI*. Penguin Random House Books. https://www.penguinrandomhouse.com/books/741805

We mean more mundane AIs, like domain-specific agents, SLMs, and fine-tuned LLMs trained to augment everyday people in everyday tasks. Sometimes these agents "wake up" once a week, complete a critical task for your team, and then "go back to sleep". Sometimes they operate ad hoc or on demand. Sometimes they operate continuously in the background. It depends on the business use case.

By focusing on the mundane, leaders become better attuned to the dull and time-consuming tasks that lead to "quiet quitting" and customer churn. Making these everyday problems concrete and elevating them to the status of "important enough to spend time and money solving" helps to create a workplace culture that is more open to AI automation; employees more readily trust AI when they feel that the AI is interested in making their lives easier.

Fundamentally, these agents should not be thought of as merely models that benefit the business (for example, better product recommendations), but instead as a core unit for business success. At the risk of anthropomorphizing agents, a cohort or portfolio of intelligent agents must be recruited, onboarded, trained, rewarded, promoted, and directed at solving business problems, much like their human counterparts.

This may sound far-fetched, but as the science fiction writer William Gibson posited[4], "the future is already here, it's just unevenly distributed." Many organizations already use intelligent agents. For example, if your software engineers use Github Copilot to build and test software, they are using an intelligent agent for efficiency gains.

Commercial-off-the-shelf (COTS) solutions like Copilot can be very valuable, but they are not a fundamental differentiator because any business can "employ" these agents[5]. Differentiation lies in domain-specific intelligent agents trained on your data and for your business use case. These intelligent agents will become foundational to business impact,

[4] https://en.wikiquote.org/wiki/William_Gibson

[5] Moreover, there is some question as to whether generalized AI is sustainable, even for Big Tech. A recent Wall Street Journal report suggests GitHub Copilot loses an average of $20 per user per month due to the app's high operational and compute costs. See "Big Tech Struggles to Turn AI Hype Into Profits" by Tom Dotan and Deepa Seetharaman, October 2023. https://www.wsj.com/tech/ai/ais-costly-buildup-could-make-early-products-a-hard-sell-bdd29b9f

from significant efficiency gains to re-engineering processes to new growth opportunities. Customized intelligent agents will become a strategic differentiator and source of economic moat for businesses.

So the question isn't "What can I do with Gen AI?", but "How can Gen AI support my organization in solving a business problem"? If the problem is related to efficiency, then how can GenAI help me lower my cost basis in the function? How can I re–engineer a function - marketing, HR, customer service, legal, etc - with the support of GenAI tools? Or the business problem could be about growth and opportunity. How can GenAI tools help improve existing products or generate new ideas for products and services?
The point is to focus GenAI on use cases critical to business outcomes that ultimately impact your bottom line. Smart companies are capitalizing on these advancements in AI to not only empower their employees but also unlock new possibilities for the business. This is a generational opportunity for leaders to shape the future of work and AI together.

This is Your Sign to Start Now

Imagine a world where your interview candidates, after hearing about the team they'll be joining, ask "and tell me about the intelligent agents that will support me in this role."

There's an old proverb that goes "the best time to plant a tree is 20 years ago; the next-best time is today."

We understand that there's a lot of noise out there now, and so much hype. This book is not a sales pitch for any specific tool, product or service. Instead, we offer it up as a guide to help you on your journey to become proficient in employing intelligent agents to augment your workforce.

We will ask you to look inward at your own organization's existing practices and data (Chapter 1), to compile business questions and evaluate use cases (Chapter 2), to plan for how to get AI to production (Chapter 3), and to build a culture of AI solutions development across your organization (Chapter 4). Finally, we provide you with several blueprints for AI solutions (Chapter 5) which we have found useful in our own work, and which may work well for you, too.

We have learned to think about custom AI models like interns – energetic and a bit clueless on their first day of work, but who through experience and nurturing will grow to contribute. Likewise, we evaluate off-the-shelf AI solutions the way we interview job candidates – how will you augment, energize, and support our team? And just as we would never hire a candidate, however brilliant, who showed disdain or disinterest in our team or our goals, no AI tool is worth the alienation of our talented humans, who are as Ronald Coase explained, at the core of value generation.

Two Important Leadership Considerations

As leaders begin building AI capabilities, two additional shifts in mindset.

1. From Deterministic Compute to Probabilistic Compute

For the last 50 years, as companies have applied technology for productivity and growth gains, the compute and applications have been deterministic. This means a computer does exactly as its human counterparts tell it to do, including making mistakes (called bugs). Compute outcomes were known and 100% predictable, assuming no bugs. You can think of logic gates opening and closing in programmed patterns, often at scale.

In contrast, gen AI and LLMs are probabilistic by design. Given a training data set and/ or prompt, they produce outcomes that are presumably statistically accurate, but not always. We are familiar with the notion of "hallucination" or fabricated responses. This rightfully gives pause to many leaders. On the one hand, gen AI and LLMs offer the capability of prediction at scale, which can massively benefit organizations, as we discuss later in the

book. On the other hand, there is inherent risk in getting predictions wrong or producing inappropriate or factually incorrect content. Gen AI and LLMs act more like humans than computers, so guardrails and internal controls are required. One way to proceed is to consider high stakes vs low stakes use cases.

2. High Stakes vs Low Stakes

When considering AI enablement, thinking through high stakes and low stakes implementations is a useful framework. High stakes implementation involves use cases that may impact financial transactions, customer experience, and/ or reputation, particularly with external stakeholders. Low stakes typically involve internal uses for information retrieval, business intelligence, and automation.

A concrete example is helpful. When planning a vacation, AI tools like ChatGPT are effective at creating itineraries or agendas, generating ideas for the experience. However, you would not yet delegate airline ticket purchases or hotel bookings to an AI agent because a mistake will have significant consequences, financially and experientially. The AI agent also cannot fully account for context or preferences. The act of delegation is important - what are you willing to delegate to your budding “interns”/ AI agents?

The key take-away is to start with focused internal use cases and begin building trust and experience with AI systems. When you do, your organization will build new capabilities to leverage your talent, differentiate your products and services, and open up new revenue opportunities.

Chapter 1: Defend Your Moat

In Q2 of 2023, on the heels of the Silicon Valley Bank collapse, the tech community erupted over an internal memo[6] leaked from Google expressing fear that they are losing the AI arms race. “We have no moat,” it read, going on to suggest that companies like Google and OpenAI cannot outcompete the breakneck pace of open source innovation.

The memo’s reference to the threat of “open source” is almost certainly a nod to HuggingFace, a new and thriving platform hosting open source models (as of the time of this writing, there are over half a million) for mostly unrestricted use, including for-profit.[7]

The availability of open source models is good news for the tech startup and enterprise modernization communities, and there has never been a better time for building

[6] “Google ‘We Have No Moat, And Neither Does OpenAI’” by Dylan Patel and Afzal Ahmad (May 2023) https://www.semianalysis.com/p/google-we-have-no-moat-and-neither
[7] https://huggingface.co/models

home-grown in-house solutions, so long as you remember “Schmierer’s Rules of AI Monetization”

Schmierer’s Rules of AI Monetization

The following rules are named after Rotational’s Chief Operating Officer, Edwin Schmierer, the one who is always making sure we stay on track with our AI builds. They are distilled from the approaches we are seeing leaders take to maximize their ROI on data science, future-proof their organizations, and deepen their moats in 2024 and beyond.

Invest in Implementing Domain-Specific Models

Here’s a fact: Open source models are rapidly converging on proprietary models in terms of performance and cost. At the same time, in the last 6 months, ~$1 billion in VC funding has flowed into open source platforms such as Hugging Face, Together AI, Replicate, and OctoML to support open source model builders. Mistral, an open source model provider, alone raised $415m. These are strong market signals.

What are the motivations to consider open source models? Organizations want to: (1) control their destiny/ avoid platform dependencies on proprietary models; (2) protect their privacy and data; and (3) customize models to their unique use cases and requirements.

Leaders and data teams must start implementing domain-specific LLMs to understand their benefits and costs. Practically speaking, this means examining LLM use cases, researching open source models, and gaining experience with the technique of transfer learning (see below). Teams that create structured, iterative, and repeatable processes building and deploying LLMs will generate significant value.

Deepen & Augment Your Data Strategy

One common refrain we hear: “We don’t have enough data to get started”. This is unlikely to be true. Transfer learning is the practice of using a pre-trained model on a new machine learning task that has a lot less data. Data teams have the convenience of bypassing initial model training steps and recomputing one or more layers at the end of a neural network for a specific use case.

If you have industry experience and expertise, you most likely have enough first-party (proprietary) data to get started. Leaders should seek to augment with 3rd-party data sets and real-time sources from their industry or adjacent industries. Data quality, diversity, and provenance matter much more than big data and organizations that learn to create and apply meaningful corpora, data sets, and sources will accelerate their growth.

Be Strategic with "Exploratory Compute Time"

Cloud service providers like Amazon, Google, and Microsoft are betting on offering hosted AI compute resources in a world where everyone is training bespoke models. While cloud GPUs and TPUs may not seem terribly expensive at first glance (a dollar or two per hour), machine learning is by definition a very experimental process, so they add up. While we encourage the development of in-house tooling, we recommend being strategic about budgeting time and resources for what we call "exploratory compute".

Many of the hosted data analysis platforms that came to prominence over the last decade did so on the promise of unlocking the potential of our data by making it easier to materialize and visualize it to perform exploratory analyses. The implicit hypothesis was that by making computational techniques like map-reduce and supervised machine learning more convenient, in-house research would accelerate, and actionable insights would emerge organically.

In practice, many of the leaders with whom we speak have realized little value from such tools. While these solutions do make machine learning concepts more accessible by furnishing sandbox environments and low-code analytical tools, they almost never lead to the effective delegation of labor or decision-making to a model. Moreover, using these tools often requires significant cloud migration work to be done (by humans) first, which not only increases operational and storage costs, but introduces lag.

When it comes to spending, we must also remember to measure the things that actually matter to our organizations, and not to get distracted by the metrics used by academics and research labs pushing the boundaries of AI in pursuit of artificial general intelligence (AGI). This is surprisingly difficult, even for seasoned experts like us. In October 2023, Together AI, a GenAI cloud platform, released a training data set consisting of *30 trillion*

tokens curated from 100 billion documents.[8] In February 2024, Google released its latest foundation LLM called Gemini 1.5 Pro with a context window of 10 million tokens.[9] One year ago, ChatGPT's maximum context window was 16,000. Suddenly we were all talking trillions, not billions, of tokens.

But unless you're one of those tech giants or unicorns with limitless resources for exploratory research, most of us will have better luck measuring AI effectiveness in hours saved and dollars made than in token counts and context windows.

Be Data Agile

When it comes to AI/ML, data preparation is project-specific. Unfortunately, this means AI/ML projects nearly always require a focused corresponding data engineering effort. This can be confusing and frustrating for leaders, especially if your organization has already spent a lot on data modernization and migrations.

The "Data Agile" Approach

Software engineering has benefited tremendously from agile methodologies: roadmapping, sprint planning, continuous integration/ deployment (CI/CD), etc. The same is true of machine learning, data science, and AI.

Data agility means teams can maximize the time value of data - the idea that data has its greatest value when it is first generated (and erodes gradually over time).

New tools and databases like Ensign[10] can enable secure data collaboration while supporting data augmentation strategies and boosting data quality. Data agile teams use these tools to reduce time to insight for users while protecting privacy and maintaining security practices.

Organizations that adopt agile data engineering practices and tools will be in the best position to develop new capabilities and opportunities afforded by genAI and LLMs.

[8] https://www.together.ai/blog/redpajama-data-v2
[9] https://blog.google/technology/ai/long-context-window-ai-models/
[10] Ensign is a secure data collaboration tool and event sourcing database currently in beta at https://rotational.app/

Below are some reasons an AI/ML project could introduce new data engineering needs.

- Organizational data may be sparse or insufficient. Models learn by example, and if there are only a few hundred representative examples available, it may be necessary to collect or generate more data before AI/ML work can begin.
- Organizational data may be available but not readily accessible. This can be the case if your data are stored by a vendor, or firewalled, or siloed across the organization.
- Organizational data may not be labeled. Training a model to take an action means providing the model with a historical record of events and the actions that were taken as a result. Unlabeled events must be labeled before model training can begin.

We must resist the alluring but false notion that merely aggregating all our organization's data into a single repository or lake will automagically unlock understanding and value – that's not how it works. Instead, encourage your organization to take an agile, modular approach to data engineering.

In most cases, it is far better (and cheaper) to allow your AI and analytics use cases to drive your data engineering initiatives rather than to treat them as independent or sequential processes.

Ship More Models

"If it's not in production, it doesn't exist." We heard this directly from a senior product manager responsible for deploying models in an insurance startup disrupting the industry with real-time machine learning and automation.

The organizations that are best suited for 2024 and beyond are those with a singular focus: ship more models. That means getting more value and insight from unstructured data and generating valuable feedback loops to improve models and deepen your moat. The longer you wait, the less experience - and data - you'll have.

Now is not the time to sit on the sidelines, but to act. Your competitors already are.

What is My Moat?

So Google thinks you have a moat. How do you figure out what your moat is?

It is usually easy to identify at least one moat in the course of our initial discussions with a customer. Oftentimes customers are surprised by these moats because they are so rooted in the mundane. In our experience, your moat is likely related to your organization's domain data and domain expertise.

Consider a hypothetical organization with a team of investigators who have, over time, found clever ways of narrowing the scope of an investigation or a deep understanding of a core set of data sources. While these data sources may be very messy and unstructured, the team is smart and has developed ways of leveraging the messy data. This is a form of domain expertise that many employees are conditioned to dismiss because it feels too mundane to see as an intelligent strategy.

Another common pattern is institutional shame about data quality. We hear so many organizations deride their in-house data as "garbage in, garbage out." Often the problem is not that the data is messy or "garbage", but that the data is unstructured (transcripts, documents, notes, etc). Unstructured data is more challenging to work with than structured data, but it often encodes valuable information and patterns that can be leveraged by intelligent agents.

In the next chapter, we discuss strategies for spotting AI use cases in the wild, which can also help to accentuate the moats worth defending at your organization.

Chapter 2: Spotting Use Cases for AI

One of the hardest things to do is to learn how to spot good use cases for ML/AI in the wild. Instead, executives and engineers tend to fixate on specific solutions and tools. Taking a solutions-first mindset requires us to resist this impulse. The most successful AI projects start with a use case in mind.

In our experience, it works best to look inward and find a small, but real pain point within your organization. While these may not always appear glamorous at first, remember that the purpose of intelligent agents is to augment and support your talented humans.

There are some telltale signs that you learn to look for that signal an opportunity to solve a problem with data: employee turnover, customer churn, repetitive review tasks, tedious

data entry duties, long onboarding processes, etc. In a word, these are places where introducing an intelligent agent could help either reduce friction or "automate the boring stuff" or surface hidden insights.

Let's take customer service as an example. Despite all the innovations in customer service such as self-service systems and chatbots, many customers have negative experiences because the automated solutions do not meet their needs. How many times did you input all of your information through an automated system only to have a customer service representative (CSR) ask you for the same information?

Imagine instead if your CSRs had some knowledge about the customer's history with your business that will enable them to provide a more tailored experience? Think about how much better that interaction will be and now instead of having an unhappy customer you have a customer who is delighted and is more likely to recommend your company to others. This leads to a virtuous cycle leading to enhanced customer experiences that create the kind of enthusiasm for your product that ultimately helps grow your business. As your business grows, the CSRs handle the demands better than ever, boosting the number of happy clients, who will start to champion your products.

Low-Hanging Fruit for Automation

Language models are very good at automating tasks that require a person to write, review, or evaluate a lot of documents. They are also very useful for building applications that extract value from unstructured text. An emerging (and useful) pattern in AI solutions development is to take one the half-million models on Hugging Face, perform fine tuning using a proprietary dataset, and leverage the tailored model to automate part of a business function. Techniques like transfer learning and fine tuning are effective for bootstrapping domain-specific models, even without much data.

The table below illustrates some of the most common use cases for AI we see in typical organizations, along with the indicators and sample impact metrics for evaluating the effectiveness of an intelligent agent.

Use Case	Clues/Red Flags	Sample Impact Metrics/Outcomes
The Needle in the Haystack	Employees struggle with document search and retrieving information from company knowledge bases and wikis.	New employees quickly become productive and onboarding time drops by 50%.
The Ever-Overflowing Inbox	The team is inundated by routine document processing, review, and approval tasks that can't scale due to their reliance on human analysts.	The model processes the majority of documents and only 1 out of every 10 needs to go to a human analyst for review.
The Quiet Quitter-Maker	Boring, rote tasks lead to mistakes as well as employee disengagement and churn.	Employee retention rate doubles in the first year of deployment.

While every company is unique, these pain points tend to be fairly consistent across industries from healthcare to cybersecurity, and our team has seen a lot of innovative solutions. We've even been lucky enough to implement some of them!

Use Cases by Domain

Applying a functional lens can be helpful. The basic framework is to evaluate workflows in each function and determine the application of LLMs or GenAI tools to solve a real business problem. A common theme in these use cases is the ability for LLMs to unlock the value of unstructured data, often living in your knowledge base or data systems, at scale, opening new possibilities for visionary leaders.

Marketing & Sales

- **MarComm agent**: Does your marketing team have difficulty maintaining a consistent tone, message, and appearance for marketing collateral, sales presentations, and social media campaigns? Is your marketing team backlogged with pending approvals, potentially slowing sales? Build and deploy an LLM trained on your marketing materials and brand book. Give access to your employees to run their materials through your evaluator to offer revisions in tone, quality, or messaging. Impact metrics: Productivity & Brand.

- **Product feedback agent**: Does your organization collect feedback on its product and services, only to have the data sit until the next quarter? Build and deploy an LLM trained on customer feedback to identify trends, score responses, and rank product improvements. Impact metrics: Revenue generation & Customer Experience.

Human Resources

- **Voice of the Employee agent**: Does your HR team conduct employee satisfaction and engagement surveys but often struggles with identifying patterns and actionable insights in a timely manner? Build and deploy an LLM trained on employee survey data to score responses, identify trends, score responses, and rank product improvements. Impact metric: Employee Retention.
- **Onboarding agent**: Is employee onboarding critical to long-term retention of employees? Often new employees don't know who or what to ask. Train an LLM on your company docs, org charts, policies, and advice from long-term successful employees at your organization. Make it accessible via a chatbot. Impact metric: Employee retention & productivity.

Customer Service/ Success

- **Customer Service agent:** Do you sell technical products that require consistent support from the customer service team? Does your backlog of customer service tickets only increase? Implement a domain-specific LLM trained on your documentation and previous issues. Customers interact with your customer support chatbot for instant, 24/7 support to handle the most routine inquiries, freeing up the customer support team to focus on more complex issues. Fine-tune for empathy and brand tone. Impact metrics: Customer churn & employee productivity.
- **Customer Sentiment agent**: Is your business highly sensitive to customer sentiment? Use an LLM for real-time sentiment analysis of customer feedback across various channels. Identifying and addressing negative experiences, enabling your team to prioritize actions based on customer sentiment trends. Impact metrics: Customer churn & engagement.

Each use case should be evaluated by an impact metric, informing a cost-benefit analysis. An impact metric relates to improved productivity, cycle time, customer experience, brand,

quality, and/ or faster upskilling. It's also important to note that while chatbots tend to be the dominant interface, LLMs can function as excellent classifiers and auto-graders. The above list of use cases is by no means comprehensive and you may think "I could use all of these agents." We agree, and many organizations will do so . The point we seek to make is simple: Focus on your business use cases. Choose one. Then go. Once you realize the gains, you'll never look back.

Beyond Chatbots

Chatbots are back, and they're way better than the conversational agents of 10 years ago (and way, way better than Clippy). In this case, "better" means that they're more natural-sounding, better integrated with internet search, and better packaged for distribution.

However, many of us are realizing that chatbots are not always the most intuitive interface for solving language- and document-related problems. For one thing, they are not domain experts and often are not sufficiently attuned to the vocabularies, acronyms, and other domain-specific linguistic patterns that organizations use. Techniques like Retrieval Augmented Generation (RAG) and tools like vector databases (e.g. Pinecone, Chroma, etc) are designed to correct for these deficiencies. Before going down the AI integration path, it can be helpful to think through the extent to which chit chat functionalities will effectively support domain-specific tasks, or if an alternative user interface might be more intuitive for your team.

A Note on Data Privacy, Security, and IP

The data community is starting to put more attention and thought towards data sourcing and data quality assurance. Data privacy is starting to become a common feature in OTS analytics solutions. Security has not become of serious concern yet, but we will likely observe breaches (i.e. prompt injection attacks[11]) this year that will draw more attention to this important issue. It would not hurt to be ahead of this curve.

There is also growing concern about allowing access to valuable proprietary data to purveyors of cloud-hosted services and off-the-shelf tools. Numerous generative AI

[11] "Securing LLM Systems Against Prompt Injection" by Rich Harang (NVidia Tech Blog) Aug 03, 2023. https://developer.nvidia.com/blog/securing-llm-systems-against-prompt-injection/

companies are currently embroiled in copyright infringement lawsuits, accused of using data improperly.[12] The jury is out in terms of what precedents will be set in these cases, but it is a fair bet that without the benefit of the millions of training documents and images gathered in less-than-ethical ways, many off-the-shelf GenAI tools will become much less valuable.

[12] "Generative AI Has an Intellectual Property Problem" by Gil Appel, Juliana Neelbauer, and David A. Schweidel (Harvard Business Review) April 07, 2023. https://hbr.org/2023/04/generative-ai-has-an-intellectual-property-problem

Chapter 3: Scoping and Estimating AI/ML Projects

The popularity of ChatGPT brought large language models (LLMs) to the forefront, and many companies are interested in costing out an initial project.

Many questions we hear are to the effect of: *Is this something that will take us 3 months, or 1 year? Will this cost under $50k? Is that even the right order of magnitude?*

One of the chief problems with scoping and estimating AI is that machine learning involves significantly more uncertainty than mainstream engineering tasks. As off-the-shelf AI becomes more prevalent, there is a growing misconception that AI/ML is now fully commoditized and "safe" from a business perspective (i.e. predictable, de-risked, etc). But unlike database or website development, machine learning requires the scientific method, which is inherently about risk.

Whether stated explicitly or not, all machine learning projects begin with a hypothesis (e.g. "We believe it is possible to train a model to identify high-risk events using the last 18 months of data labeled by our human analysts"). As is true in all scientific endeavors, the null hypothesis is always possible.

What do you do if your hypothesis turns out to be wrong? While this is a risk, in our experience it's *far more common* for organizations to be caught off-guard and stall out in the face of routine uncertainties (e.g. class imbalance issues, mediocre F1 scores, etc). It's also very common for different stakeholders in the organization to have different hypotheses, and different expectations about what success will look like (this is discussed more in "Getting to Production" below).

It's important to get good at communicating and interrogating AI/ML hypotheses as early in the process as possible. Using this strategy, many poorly conceived hypotheses can be abandoned or reformulated with a minimum of time, money, and effort. Below you'll find a list of some of the most common problems with machine learning hypotheses that we encounter.

AI Hypothesis Deal-Breakers

- Automation action is something that a human cannot do.
- Automation action is poorly defined (the model should "just *know* our domain").
- Automation action is something that requires a human (e.g. for regulatory or sociocultural reasons).
- Historical records of human-driven actions have not been recorded.
- Historical records of human-driven actions are not machine-readable (e.g. hand-written documents).
- Historical records of human-driven actions are insufficient (e.g. only 3 months of data).

After the hypothesis has been socialized and found viable, our next strategy is to sketch out a roadmap in the form of a Data Product Requirements Document (DPRD), which ensures that the business and technical teams are aligned on the product's purpose and scope.

Writing Data Product Requirements Documents

A Data Product Requirements Document (DPRD) defines the requirements for the product that your technical team will be building. Our template is a twist on the traditional Products Requirements Document (PRD), tailored specifically for machine learning products.

A DPRD does not have to be overly complicated. In fact, it's best to keep it to as few pages as possible so that it is easy to reference through the course of product development. The following is an outline of what should be included in the document.

Purpose

It is very important to ensure that everyone has a shared understanding of what the end goal is. The end goal dictates the type of LLM(s) used and the metrics required to evaluate them. The purpose should include an **estimated impact metric** to ground it in business use cases and outcomes. For example, if the desired outcome is to reduce customer churn, then define a relevant impact metric.

Data Sources

It is also critical to define the data sources used to build the LLM. This can be either internal data or third party data or a combination of both.

Architecture Diagram

The architecture diagram serves as a guide to show the data flows from data ingestion all the way up to the end user interface where the model predictions/outputs are served. This doesn't have to be anything fancy. It is also not advisable to prematurely commit to a tech stack.

Dependencies

Dependencies include everyone who is involved in product development and domain experts, as well as external dependencies such as external personnel, tools, or software libraries. It is important to identify these in order to understand the costs and risks involved.

Evaluation Metrics

Evaluation metrics provide a guide to help evaluate the performance of the LLM(s). Evaluation metrics are not the same as impact metrics. Evaluation metrics measure LLM efficacy while impact metrics measure business outcomes.

Feedback/Monitoring Framework

Once the metrics are defined, they need to be part of a monitoring framework that will be used to continually monitor the performance of the LLM. It is not sufficient to just use metrics during the model training phase. Model drift and data drift cause models to go stale after they are released and so it is essential to have monitoring in place. There should also be a mechanism to capture and use feedback to continually improve the product over time. In fact, feedback is even more important than metrics because feedback from end users is what can provide the best insight into where the model is weak and the areas that require improvement.

A sample DPRD can be found in the Appendix at the end of this book.

Pricing

Once you have developed your DPRD, it will be easier to estimate the cost of the effort. In the table below, we can see a hypothetical break-down of the different phrases of a modest AI/ML effort to detect customer service emails that should be flagged for legal review.

Phase	Team	Time
Evaluate open source BERT model variants from HuggingFace	1 data scientist	2 weeks
Extract data from the support email inbox to get records going back 3 years. Extract data from in-house database tables that preserve records of actions taken by the support team during the same time period. Join email and database data together.	1 data engineer	2 weeks
Vectorize and wrangle the data to prepare it for machine learning	1 data scientist	2 weeks
Perform fine-tuning using the aggregated data to attune the base model to the organization's domain-specific language.	2 data scientists	4 weeks
Evaluate preliminary results with domain experts at the organization and get feedback.	1 data scientist + 1 analyst	2 weeks
Design a Python API to wrap the fine-tuned model	1 machine learning engineer	2 weeks
Develop and deploy an internal Streamlit dashboard to enable the team to continuously monitor the model.	1 machine learning engineer	2 weeks
Deploy the API using Microsoft Outlook and AWS.	1 machine learning engineer	2 weeks

A secondary but non-trivial cost is the compute and storage resources required to develop and deploy your model. The cloud pricing calculators tend to be very complicated and the risk of spiraling costs is real.

Many organizations opt to set up dedicated services such as SageMaker or Azure ML and in fact, setting up these services is the bread and butter of many AI consulting firms; but this is the path to surprise costs and the GPU minutes add up quickly (see Chapter 1 discussion on curbing "Exploratory Compute")!

Depending on the engineering expertise of your team, a hosted cluster of 6-12 Kubernetes nodes can provide a complete ML workflow from hosting Jupyter notebooks to MLFlow for training and tuning to API and UI services for deployed models. If you have the devops expertise, this is often the most flexible mechanism but will cost you anywhere from $2,300 - $5,800 per month depending on the number of disks, nodes and their instance sizes. This monthly cost is equivalent to the one time cost of buying a gaming desktop with 2 GPUs – it's no longer in the cloud, but it is hugely cost effective to get started.

Another complication of pricing (also true in the case of more traditional engineering tasks like website development), is that novices typically tend to require more time than a seasoned engineer. Many organizations at the beginning of their first AI/ML effort, or those without in-house machine learning practitioners, might benefit from working with an external team of experts in a consultative capacity to keep things on track.[13]

Make sure you understand your plan of action for monitoring and controlling these costs before you start your project.

Getting to Production & Last-Mile Engineering

The critical point is to get intelligent agents into production as quickly as possible. Getting to production is critical because it is only then that your organization will gain the experience and realize the impact of using intelligent agents.

However, the discipline of MLOps is still new, and not all organizations think of "deployment" or "in production" in the same way. The definitions and expectations that make sense for a cloud-native startup may not be the same as those at a mature enterprise software company. It may be necessary to develop an internal standard for what this means to you and your team.

[13] To see how we at Rotational cost out different types of AI projects, see our Services page at https://rotational.io/services/

It may help to ask yourself the question “what level of packaging is necessary for the AI to actually act as part of our team?”

Here are a few of the different ways we have observed organizations conceptualizing the end-result of an AI project.

- A fully autonomous agent deployed using Docker and Kubernetes and composed of multiple networked microservices coordinating different phases of the automation process.
- A scheduled Databricks job that “wakes up” and runs a Jupyter notebook once per week, writing the results of the model back to a database table.
- A code repository containing a Python API that engages serialized models stored in S3.

Chapter 4: Building the Right Team

VentureBeat reports that 87% of data science projects don't make it to production.[14]

Why? As discussed in the last chapter, a large number of data science efforts fail because the underlying hypothesis isn't viable. But there is also a cultural problem when it comes to machine learning – and it's not what you think.

Most executives expect resistance from their more seasoned employees, anticipating fears about AI adoption related to being micromanaged by disembodied algorithms or even automated out of a job.

We're more worried about the data science and AI/ML communities. Many data scientists simply aren't interested in building solutions.

The Research-Solutions Gap

The first ten years of data science (roughly 2010-2020) were focused primarily on research. The so-called FAANG companies poured hundreds of millions of dollars into building machine learning research teams, and enterprises from banking to consumer electronics promptly followed suit. The resulting teams were encouraged to produce research publications and open source libraries, and to evangelize their organizations as data-driven and forward-thinking. Putting models into production was not top of mind.

The next ten years of data science will be about building solutions. They will be about using AI/ML not for self-promotion, but for economic survival.

[14] "Why do 87% of data science projects never make it into production?" VentureBeat (July 19, 2019) https://venturebeat.com/ai/why-do-87-of-data-science-projects-never-make-it-into-production/

Unfortunately, the data science programs that emerged over the last ten years have yet to pivot. Most of these educational programs and bootcamps emphasize algorithms over engineering, and many students graduate without ever building an end-to-end solution.

Also a holdover from those first 10 years, machine learning research is often seen as having more of a "cool factor" in the data science community. As such, there's very little coverage at data science conferences and meet-ups about use cases or AI project management, widening the gap between data science methods and applied solutions development.

The recent emergence of "MLOps" is also indicative of the research-solutions gap; while the title "data scientist" increasingly refers to budget-friendly data analysts, the rise of the more highly compensated "MLOps Engineer" signals the market's need for seasoned machine learning practitioners who understand what it takes to move a research project to production.

When building a data science or AI/ML practice at your organization, it's crucial to stress that solving problems is more important than training models. Product-oriented data science teams need to limit their "exploratory compute time" (see Chapter 1) and start developing their MLOps, data engineering, data management, and software engineering skills in earnest.

Breaking Down Silos

The more dependencies and barriers to deployment, the less you'll ship. These barriers often show up as database silos, specialization silos, and team silos.

Leaders should work to end messy hand-offs where data teams expect to delegate the productionizing and deployment of their models to separate teams like devOps or MLOps. Instead, allow data teams to own the end-to-end process from exploratory data analysis to deployment and scaling and monitoring. New tools and databases exist to accelerate the process, no matter your cloud provider. Empower your data teams to explore these new tools and come to you with suggestions for how to break down the silos that throttle AI solutions development.

Red and Green Flags of Job Interviews

For those leaders out there who are building a team from the ground up, we'd like to share some of the red and green flags we've learned to spot in interviews.

We have developed some good interview questions (see the table below) that have helped us to build a team that has what it takes to get models to production.

Question	Green Flag	Red Flag	Why?
"Have you used Technology X?"	"No, but that would have been *such* a good way to solve this semantic search problem we were facing at my last role! We were trying to..."	"Yes, I have a Double+ Amazon Wizard certification in Technology X. I also have certifications in Tools Y,Z,Q,L and P. They are great."	Look for people with firsthand experience with the problem the tool is designed to solve (even if they don't have direct experience using it yet).
"How about GitHub?"	"We used a few private repos at my last job to make sure the analytics API stayed backwards compatible and passed through CI/CD as we added more models."	"I mostly use Jupyter Notebooks because that's easier. Usually the devops people can just zip them up and deploy them in Databricks or something."	Look for people with respect for engineering. At the very least, you don't want to bring someone on who is going to require their own MLOps assistant.
"We're currently on prem, but we want to migrate to NooCloud. Thoughts?"	"Oh, that's great. Fortunately the migration won't block us from getting started, since we'd be hitting an in-memory database at first..."	"Are you kidding! NooCloud is absolute garbage! Their UI is trash and their consistency guarantees are a joke."	We like a data scientist who knows a bit about compute and storage. Trash-talking techies are a major red flag.

The ideal candidate can talk about mundane, everyday applications for machine learning. They won't enumerate dozens of cloud certifications when you ask about how they would approach building one of the intelligent agents described in Chapter 5. We look for candidates who understand the realities of data science product development, like scoping and story estimation. They can think in terms of timelines, and they have ideas about how to pivot if an initial hypothesis doesn't pan out.

Perhaps most importantly, they have enough empathy and collaboration skills to adopt the Engineering department's software development best practices and to be nice to your domain experts. One of the tech-sphere anti-patterns we try to avoid at Rotational is the all-too-common trap of denigrating other people's code. Perhaps it's because technologists tend to be perfectionists, or because critique is a convenient way to demonstrate engagement, but for whatever reason, there's often much more disparagement in the tech space than there is praise. For us, the ideal candidate doesn't use the trick of trash talking to make themselves seem smarter. They have seen people make good and bad technology decisions in the past (maybe they have made the mistakes themselves!), and now they know better.

Chapter 5: Templates for Success

We know abstractions and rules of thumb only go so far. What helps most of us learn to be successful is seeing examples of success in practice.

In this chapter, we'll explore some templates for bringing intelligent agents onto your team based on successes we have seen firsthand. We'll "meet" three hypothetical agents: a Customer Service Rep's Assistant, a Liability Monitor, and an Onboarding Buddy.

We'll walk through each of the stages as described in this book and illustrate how common AI components (chatbots, LLMs, RAG, knowledge graphs, etc.) can be ensembled together to produce these intelligent agents. For each of the three examples, we'll provide an anonymized scenario and talk about the pain point/use case signals described in Chapter 2. We'll outline the underlying hypothesis as proposed in Chapter 3 and indicate team members needed to get to production as described in Chapter 4.

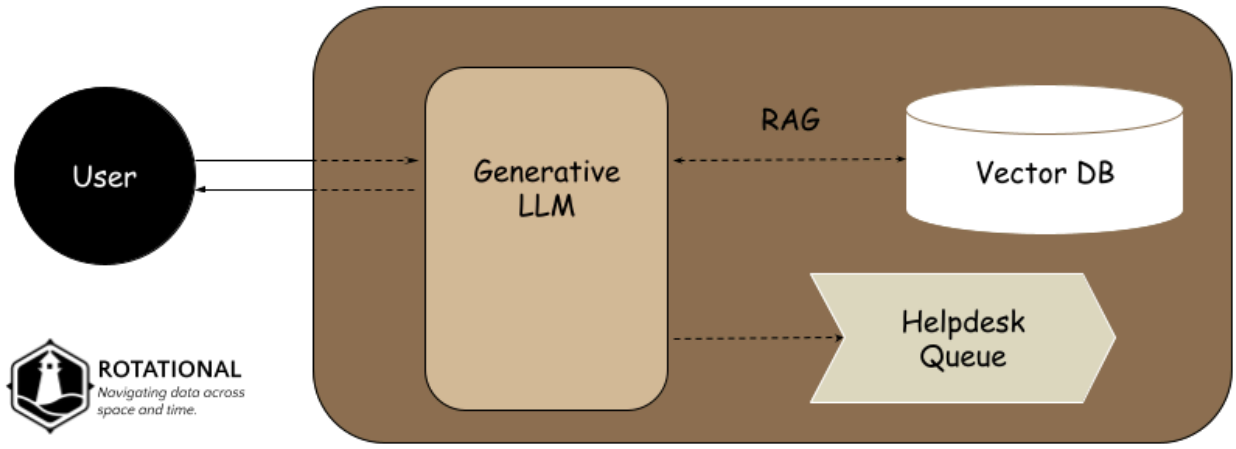
The Customer Service Rep's Assistant

Consider the help desk problem referenced in Chapter 2, with the Customer Service Representative who is forced to ask an already-frustrated customer to repeat information she provided in a previous call. For this hypothetical scenario, let's imagine our customer is calling about to report a bug in a piece of enterprise software and request a fix.

The pain point signals are **data entry** and **repetition** – the customer has already provided the necessary information on prior calls, and it is bad customer service to ask an already-frustrated person to do something tedious like repeat themselves. Moreover, our Customer Service Rep will likely have to perform unnecessary data entry, duplicating the bug report that was already captured.

Our hypothesis is that we can help our talented CSR formulate factual, relevant responses in real time using Retrieval Augmented Generation, a vector database of prior bug reports and resolutions, and a generative LLM.

A high-level architecture diagram for this agent might look something like this:



At a minimum, this project might require:

- A data engineer to extract historic bug reports and resolutions.
- A machine learning engineer to vectorize the reports and load them into the vector database and implement a RAG API to query the vectors.
- A front end engineer to build out the UI that integrates the off-the-shelf LLM solution with the backend API.
- A customer service representative to help evaluate the model and API.

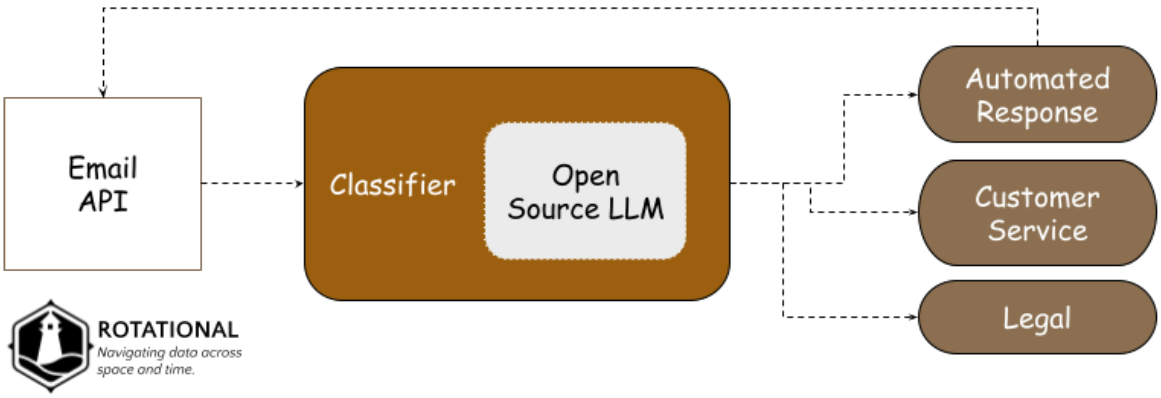
The Liability Monitor

Next, recall the monitoring agent mentioned in Chapter 3. Imagine that patrons at a popular seafood chain restaurant are able to send messages to a support@helpdesk.com account. On the other end, a support team with shared access takes turns with on-call duty, reading and responding to the messages as necessary. The team receives hundreds of emails per week, and a wide variety of sentiments, with a small proportion of negative messages indicating a serious grievance (e.g. claims of food poisoning, discrimination, etc).

The pain point signals are **document review** and **high message volume,** not to mention the risk associated with accidentally missing an important message. Our hypothesis is that we can train a model to automatically review messages sent to the shared account and identify high severity cases for routing to the legal department for special review. We

believe we can bootstrap the model by applying transfer learning to an open source LLM to enable it to perform classification.

Our architectural diagram might look something like this:



This project will likely require:

- A software engineer to determine how to use an existing API or else implement one to support message routing.
- A data engineer to help collect and retrieve historical data from the email inbox and the customer database.
- A data scientist to identify and evaluate existing language models that can be used for transfer learning, to preprocess and prepare the training data, and to apply transfer learning for the multiclass classification task.
- A machine learning engineer to operationalize the transfer-learned model and integrate it into the API.
- At least one person each from the Customer Service and Legal departments to help evaluate the model and propose appropriate thresholds.

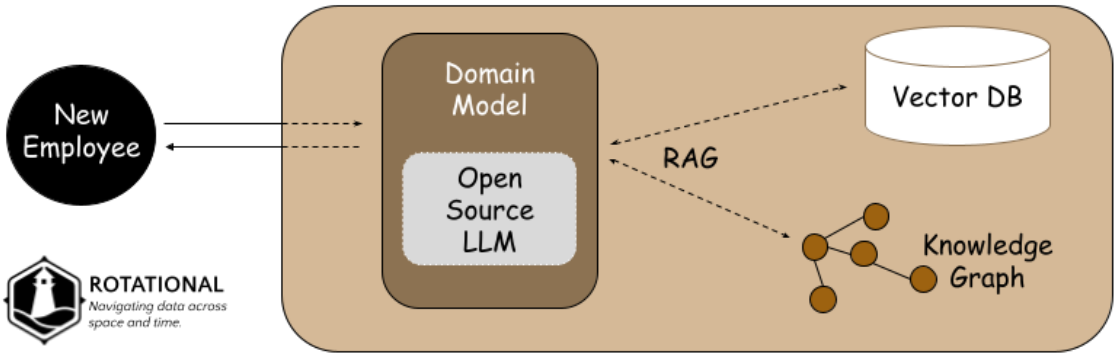
Onboarding Buddy

Finally, let's explore an example most of us can relate to – onboarding at a new job. The rise of remote work has had a big impact on onboarding processes. In the past, new employees might have had an "onboarding buddy" delegated to help show them around the physical office, introduce key team members, and assist with acclimation. Nowadays, it's common to

have coworkers that we have never actually met in person! As a result, onboarding processes now rely heavily on the new employee's independent navigation of digitized documentation, standard operating procedures, and corporate knowledge bases. Unfortunately, many employees struggle to find what they need because they have not yet learned the terminology and acronyms to use in their searches.

There are a few pain point signals here, including both **document search** and **document review**. Our hypothesis is that we should be able to combine a fine-tuned language model, a vector database, and a knowledge graph, and use Retrieval Augmented Generation to surface relevant information to new employees even if they don't use exactly the right search terms.

A birds-eye view of our application architecture might look a bit like this:



The implementation team for this project should probably include:

- A data engineer to help index all the data from existing corporate knowledge bases.
- A data scientist to identify and evaluate existing language models, preprocess and prepare the training data, and to apply fine tuning.
- A machine learning engineer to vectorize the knowledge base articles, load them into the vector database, implement a RAG API to query the vectors. The MLE will also need to operationalize the transfer-learned model and integrate it into the API.
- A few new employees who have recently gone through onboarding or are currently in the process who can help with validation and testing.

Conclusion

The intelligent agents sketched out in the previous chapter highlight the value of **contextualizing AI**. Much of the hype frames AI as uncannily universal. Advertisements imply that integration will be trivial, and that getting ROI from AI is a “no-brainer”, virtually guaranteed. We find these portrayals misleading. Problems – and their solutions – are always highly contextual. Incorporating Gen AI into business isn’t just another software implementation. It’s not plug-and-play. There are a lot of implications and risks, some known and many unknown.

Nonetheless, we hope this book has left you feeling empowered. As AI engineers-turned-startup- founders, **we think *you* have the edge**.

Here’s why. Holding nuanced contexts in our minds is where humans shine.

Knowing how overwhelmed the help desk is, how frustrated Barb from accounting feels as she onboards a fully remote team, how much of a battle it is to get anyone from Legal on the phone these days — your grasp of the complex contexts at your organization is a superpower. The complex matrix of intersecting circumstances and personalities that uniquely defines your organization is something no chatbot or genius AI engineer in Silicon Valley understands. And it’s the key to steering your organization on their AI journey.

Good luck!

The Rotational Team

The Rotational Labs Story

Rotational Labs is an AI enablement and data science services company. We're a proven team of senior data scientists and engineers with a collective 50 years of experience tackling complex data and machine learning projects end-to-end in government and commercial markets.

Established in 2021 by three long-time colleagues from Georgetown University and collaborators on Scikit-Yellowbrick,[15] Rotational's founding team set out to build an empathetic tech company capable of developing customized distributed data systems composed of networked databases, human domain experts, and intelligent agents.

The founding team: Dr. Benjamin Bengfort, Dr. Rebecca Bilbro, Edwin Schmierer

Our engineers: Patrick Deziel, Prema Roman, Beci Lambrecht, Danielle Maxwell

We help organizations make smart and responsible AI investments that accelerate growth, reduce costs, and automate the boring stuff.

And we love working with domain experts, like you.

[15] Yellowbrick is an open source, pure Python project that extends the scikit-learn API with visual analysis and diagnostic tools. See more at https://www.scikit-yb.org/en/latest/

Ensign, our intelligent agent that helps us build LLMs.

Appendix: Sample Data Product Requirements Document

Genre Detection

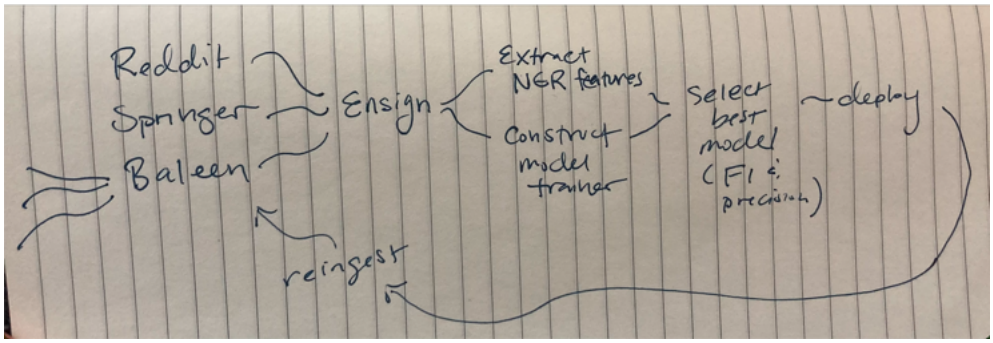
PURPOSE

To train a model to differentiate between genres (e.g. scientific research papers, newspaper articles, social media posts). It will be a fast and cost-effective way to bootstrap a charismatic language detection tool.

DATA SOURCES

1. Springer Nature API (scientific research papers)
2. Baleen (real time ingestor of newspaper article snippets via rss)
3. The Reddit dataset (somewhat preprocessed batch of Reddit posts)

ARCHITECTURAL DIAGRAM



EVALUATION METRICS

Because we will be training a classifier, we can evaluate the model straightforwardly using the F1 score. Precision will also be useful; we are most interested in a solution that can distinguish the scientific validity of a given piece of text, so we will prefer a model that can identify non-scientific text with a high degree of precision.

MONITORING PLAN

We are working to ingest more datasets of scientific papers, and ideally labeled data from predatory journals. Therefore, we will plan to train a completely new model in 3 months to replace this initial model. We will then use this initial model to help us in benchmarking the enhanced model.