Deloitte.

Deloitte AI Academy™

Generative AI Fluency
Part 1: Welcome to Generative AI

Consolidated Transcript

Table of Contents

Welcome

Hello and welcome to the first of three Generative AI workshops. So, this is part of Deloitte AI Academy's Generative AI Fluency series and is produced in collaboration with the Deloitte Technology Academy.

Learning Objectives

So, today I'll have three sections—An intro to Generative AI. So, we will go over the history and just what is Generative AI, the components of what makes up Generative AI, and solutions. So, from the hardware stack all the way up to the applications, and then finally discuss the life cycles of projects just so that you guys have an understanding of how these things work and how these things are implemented, and what you can expect from the phases that exist.

Meet the Team

Quick intro about myself; my name is Michael Luk. So, I'm the CTO of SFL Scientific and Managing Director here at Deloitte. I work primarily at SFL focuses on developing state-of-the-art R&D solutions (so, in the AI and data-driven space for our clients). Here are just the members of my team that contributed to this workshop. Please do reach out to them for questions or comments.

Topic 1: Introduction to Generative AI

Types of Artificial Intelligence

I like to start here to just level set on what is AI and how this field fills in the gaps.

So, artificial intelligence at a broad scale, is really just a form of intelligence for machines and computers. This is data processing, robotics, problem-solving, and thinking about how computers can synthesize information from the world.

Data science is a subset of AI that really gears towards looking at data and data-driven approaches. So, this includes statistics or moving averages, you know, the standard mean, and standard deviations, those kinds of things. And then also into model building.

Whereas machine learning is a subset of that where it's specifically geared towards a suite of models and algorithms, such as support vector machines and decision trees, random forests, those kinds of things. They are geared towards learning patterns from data, agnostics to solve the equations and underlying rules that are placed on it. So, you can think of machine learning as a general framework, say decision tree, that learns from patterns of data, and it produces something useful. Whereas, traditionally, you have like software engineering, which are typically, hard-coded rules, if-else statements, where you have to manually include or manually have a physics equation that will define what's happening. So, moving away from that paradigm of specific equations for each thing to more general frameworks.

Deep learning is one step further. It's just synonymous these days with neural networks where machine learning is geared towards models and algorithms that act on raw data, and deep learning is really acting and building representations itself of the data into something deeper. So, by that I mean, say a decision tree or a support vector machine, whatever rules and whatever features that you give it, that's what it's

going to be modeling its patterns from. Deep learning are going to create the patterns itself from the data sets you give it. You can learn these patterns and learn what's semantically meaning for representation.

And then Generative AI is the final step that we are going to talk about today. It's a huge field, not just within deep learning, but today we'll focus mainly on deep learning. So, you can think of things that are not deep learning in Gen AI that we won't cover today. So, sort of like physics, simulations, and robotics. These are still generative; we can generate stuff from them, but we won't cover them specifically today. We will talk mainly about this subcomponent of deep learning that's really been driving the last one, two, or three years of progress.

What is Generative AI?

What is Generative AI? On the left, we have discriminative model. Neither are the general models that we've had for the last 10-15 years and, make up majority of use cases still. The idea behind discriminative models is trying to define and try to model the data by what it is and what it isn't. And what I mean by that is if you wanted a model to learn patterns about the data, you can do things like the model will learn the pattern between a tumor and a not tumor in the image; benign cells, for example. Or, it can be used to differentiate between a cat and a dog, or in this case, a dog and a muffin, or legal documents in one category versus the second category, or types of receipts. Like these kinds of subdivisions ... the model will learn patterns to divide up the space so that more dog-like in this case is separated maximally from more muffin-like. And the idea of the model is to learn the patterns and the behaviors such that the separating boundary between them and the patterns that are learned by the model and modeled are the ones that differentiate between the classes.

So, in this case, we have like four stick-like vertical structures that are in the images and the model will interpret that as legs for us, semantically. But the model will understand that if you see that presence in the data, those four stick-like parts in the image is more intuitive for ... or more likely to be a dog. Whereas for the muffin case, if it sees things that are more spherically symmetric or that has this weird shape on the bottom, then it's more likely to be a muffin. And so, it focuses on things that are used to differentiate between classes. So, this is really great because it simplifies a lot of that logic. You don't need to understand what a dog is. You just need to understand what the differentiating factors are between dogs and muffins. So, maybe dogs have pointy ears, four vertical structures, whereas muffins are more spherically symmetric and more smooth. So, it just learns the differences. So, it's very easy to do because the difference is easy to spot, but it is also more brittle.

Because now, let's say that we have this model that understands between dogs and muffins and now you want to use the same model, same architecture for things like identifying dogs and cats, and you can't do it because the model doesn't really understand what a cat is and what a dog is. Just trying to differentiate between dogs and muffins. And so, the things that it is focusing on are actually the wrong things that exist, whereas the generative model, which we'll get to, is more holistic in its modeling. So, generative model learns the distribution of the data and it's basically modeling that data itself and forming an understanding. So, forming this sort of gradient and hill-like structure of what it means to be a dog, and what is more dog-like. And in the muffin case, the same thing in blue—like what it means to be a muffin, what it means to be muffin-like. And so, the models that classes separately, by itself, and just modeling what it is. And we can do with this, is that at the peak, it's more dog ... we go get more dog-like images, and near the boundaries between things, you can still get sort of this weird dog

Chihuahua-muffin-like hybrid that are starting to bridge the gap between the dividing line. And so, you can still create this dividing line by understanding the two distributions and saying, there's some line here that's 95% muffin 5% dog, or vice versa. But now you do so at the data representation level. And so, you understand the distribution of the data first and then you figure out like how the classes will. This is great. What you have for this is a model with data distribution and now you can sample from it. And this is where the generative part comes in. So, if you sample near the top of the green set, you can get very, very dog-like images coming up. And if you sample near the bottom, near the trough, near the dividing line between muffins and dogs, you'll get this freakish hybrid creature. And you can start inferring points between the current points. You don't have to just sample from what we've seen in the data set we have because now you can start generating new examples. Because you've got a complete understanding of what it means to be a dog, and what it means to be a muffin. That's the real differentiating factor. And the flip side is that it does require more data and more compute. And it's really just in the last couple of years that this has really been something we can do.

History of Generative AI

So, how did we get here? So, neural nets had been around since the forties and fifties. Over time, the state of the art has been relatively slow, up until maybe the last 10-20 years. In 1966, you have your first version of chatbots and then after that there's been 50 years of not too much progress, in terms of just the core generative models that we're talking about today.

One of the big things that came about is generative adversarial networks. These are pairs of neural networks that work against each other in an adversarial manner. And we'll talk more about that in workshop two. But effectively, what happens is you have a generator that generates images and a discriminator which says whether or not that image is a fake or a real. And this pair of models can basically train the generator to predict more and more realistic images and was used to train photorealistic faces and images that existed. And so, these have been around since 2013, 2014, in very limited capacities.

The next major milestone that existed is the transformer. So, the transformer was introduced in a seminal paper in 2017 under “Attention is all you need.” So, transformers are also a specific type of neural network where the model is learning from weighted averages, weighted patterns of the inputs, and learning the patterns from that data. I'll talk more about that. I'll go over the high-level in the next few slides. So again, this 2017 paper that introduced this idea of a tension mechanism where just weighted inputs really lay the groundwork for generating text. And you'll talk more about the language models where you're trying to predict the next word in the sequence all using this transformer architecture.

And the final thing that I'll mention, and that's been relatively recent in the last two years, are diffusion models. Diffusion models basically take an image, makes it noisy, and then denoise the image, and the model learns to denoise an image. And by learning that pattern, learning how to separate noise from the actual content can be used to understand the data itself. What is data and what is noise? And then be used to generate new samples. And then from there it's basically been iteration upon iteration of what we've had before. So, these things like GPT3, GPT4, nowadays, stable diffusion, DALL-E, DALL-E 2. All these things are formed with the groundwork that's been laid before it.

Emergent Properties of Large Models

But why now? So, as with all AI, I think the three main things that people talk about are the training data sets have gone huge. So, historically we've had data sets on the orders of tens of thousands, hundreds of thousands of samples. Now we're into the billions and trillions of tokens. So, trillions of tokens would mean billions of images. So, the data sets have gone huge, number one.

Number two is we're supporting larger and larger models that are used to learn patterns from the data. So, these models are trying to get into the millions and billions scale. And nowadays we're almost at the trillion mark. And some people are expecting that GPT4, in the order of trillion parameters. A trillion parameters meaning that the amounts, the number of weights, that a model can learn to represent the data. So, basically, a trillion numbers that will represent the data set and will represent the model. If you think about linear regression where you have W1+X1+W2X2 plus, so on … you have a trillion W1s. A trillion weights that exist. And so, this is an astronomical number. So, that's the data model size.

The third is the compute—the computer to run on these things. So, according to Moore's Law, now that we have like better and better GPUs, those kinds of things are really enabling us to use those models train on the large data, and it's still costing tens of millions, hundreds of millions of dollars to do so. But those three things combined, the data, the training … the data, the models, and the compute that these are training the data, are the three fundamentals in AI.

The first component that we see nowadays is emergent properties of LLMs. So, it has gotten to this point where we're scaling up the models, and it just so happens that when we look at it and look at the results, we see what's called emergent properties. A paper released a couple of weeks ago is maybe pointing holes that this might not be an actual phenomenon and might be a mirage, of sorts. But right now, it looks like that as you get to, roughly 10 billion parameters, there's an uptick in performance in arithmetic. So, these language models that we're fine-tuning or training, they just happen to have zero accuracy or ability to multiply numbers. And then when you get to 10 billion, you do nothing else. You just have the same architecture, but you scale it up. At 10 billion, it starts to tick up 10%, 20%, 30%, 40% accuracy of like being able to do this arithmetic. And it's not across just arithmetic, we have question-answering, competence reasoning. These kinds of things are starting to be emergent from the fact that we have bigger and bigger models. But as we go, the accuracies are picking up and that's not in debate.

As you get larger models, the accuracies are picking up. So, the three things, it’s really the larger model size, and then the data sizes and the compute. Coming back to it. And as we get larger models, we're getting to this phenomenon of emergent behaviors. But then the thing is that there's some papers recently that talk about the limits of training data, and you think that we're scraping the entire internet now we're into data sets of few orders in trillions. We're actually getting into the space where we might run out of data and the next two to ten years, depending on how the field goes. So, it's starting to get into an interesting space where there's a lot of data out there. We're collecting most of it nowadays it’s scraped to start the outline. What we do with that and what we do with it next and figuring out how we operate with that, as we see an increase in quality, is going to be one of the big things that we'll talk about in the next year or two.

Generative AI Benefits

So, there are these five major things to think about progressing through Generative AI and how we can use it.

Accelerating

So, the first one is accelerating workflows. So, by just writing a first draft of an email or a first draft of a proposal or first pass of code, I think, for developers and content writers, having that spark of creativity, getting you 60%, 80% way there is going to be a huge value add for everyone and an immediate thing that we can do for most people.

Automation

Moving into automation, starts to get into where you need to think about low-risk regimes, where it's okay that things are not quite perfect, where you don't exactly want a human-in-the-loop system. Things like marketing or chatbots, or virtual systems where you can, at some point, maybe defer back to human users or human experts, but really automate 80%-90% workflow for 80-90% people and think about how we do that at scale.

Personalize

Personalizing is the next one. It's like personalizing the kind of content that we're showing to people. So, things like making a recommendation or generating a new UX design, like how information is presented to you. Or even in the areas of education where some people might be visual learners, or some people might be readers that learn that way. You can generate... So, like an educator, you can generate someone to explain the task. Not just have lecture notes that is just out there.

Simulate

Moving further down the stack is simulate. So, these are starting to get into the area where the generation process and the generative models are used as a first step in a multi-step process. So, for example, creating digital twins is a good example in, say like, the ER&I space that we operate them where you're creating simulations of working models so that we can analyze what happens to things. So, data generation is another good example where we're starting to generate an ability to generate data points for things that are typically low data and typically have not much no content that exists in native form. So, you can think of things like contract issues. If you're looking to feel aspects of it, like generate content to give these examples of content. How could it be phrased differently, but also things like, generating images of people with tumors so that you can better educate doctors, physicians. All these ideas and concepts are where we're simulating data for a downstream model's entry task— education or other downstream deep learning models.

Create

And the final thing that is most state of the art is like raw generation, and raw creation of new content, and being able to leverage Generative AI to do things that you otherwise would not have been able to do and generate like completely new content. A lot of people talk about how LLMs are statistical models, and just like, paring back the data set that you have. But a lot of innovation, lot of creativity comes from interpolating and taking from different domains of data that is observed. So, I can totally imagine a space where we have new types of digital art that are Generative AI art that will enable people like myself who are completely unable to produce any painting or drawing to allow me to generate that content afresh and be able to do that is a really exciting thing that I think will spur a lot of innovation, not just in the art world, but across the spectrum.

And then thinking about how these things can be used downstream. So, I like to divide it up and see. These are some of the data modalities that exist. So, these things could range from images and

presumably, videos that will basically be able to generate any content that you want or people's faces. We had horse-to-zebra transformations like being able to generate novel content of that sort is really phenomenal and really becoming easier and easier as time goes on. Then there's things in, say, signal processing where you're generating speech-to-text or text-to-speech, where things can be conditioned on the input and then generate the output that exists. And so, one example is translation. Live translation would be a phenomenal one or music generation where you're trying to tailor music to each person. And the last one here listed is what people typically think about in terms of Generative AI, in terms of text and code. So, simulating and generating, new types of code, summarizing things, translating from text to text, those kinds of ideas coming about in that space. So, definitely a very rich area and there's maybe thousands more that will open up as time goes on.

Topic 2: Components of Generative AI

Generative AI Components for the Enterprise

I'm moving to the second part of this workshop. So, this is components of Generative AI. So, at a high level, there are five different categories, subcomponents you have to consider when thinking about Generative AI. A lot of these are similar to just general AI use cases. From the top, we have the hardware layer. So, the GPUs, the TPUs, the things are running on.

The second is training data. Thinking about data sets that exist on the internet or elsewhere that are scraped to use to train these models.

Foundation models themselves to talk about. Fine-tuning these models and they're sort of like an optional fine-tuning step. And then finally discussing some of the applications and possibilities with Generative AI.

Hardware Stack

So, at a high level, this is what a typical hardware stack looks like. At the bottom you have the raw, bare metals that exist. So, the GPUs, TPUs, and those kinds of things. Above those you have the raw coding languages, like the formal coding languages to interact with other GPUs. And you have the cloud data platforms above it for the AWS, Azure, GCPs of the world. Modern data platforms that are above on top of those. And then you have the model repositories, model hubs, and tuning mechanisms.

And then finally the application layer.

The things you need to be paying attention to, in terms of the Generative AI aspects are, one, at the hardware level, the bottom, you need to start really considering things like how do you make your matrix multiplications faster, and how do you do that at the CUDA level for the operating system level? Because now you're thinking about, 5%, 10% of your time saved on training time, that’s like hundreds of millions of dollars, is a significant fraction of what can be achieved.

The second thing you need to think about in terms of these types of Generative AI solutions is that they take enormous amounts of compute. So, running them on AWS, running them on an Azure, costs and requires like hundreds, thousands of GPUs out there. And being able to manage them on a distributed system is something else you need to fundamentally pay attention to. For someone like OpenAI or even ChatGPT, running per day is latest estimates put it like at $700,000 per day. So, you can imagine being

able to orchestrate, being able to be fault tolerant and resilient becomes a bigger and bigger deal for generative applications just because of the sheer magnitude of costs and scale that's required.

Then you move into modeling and frameworks, and I have a section on foundation models, the fine-tuning and training frameworks that exist out there. So, you can think about things like Megatron, DeepSpeed, or nowadays like LangChain that fit around the modern pipeline is really starting to think about like how do we operate on the output that's generative and how we do that in an efficient manner so that we can talk to the bare-bones CPUs and GPUs out there in an efficient manner.

And the final application there, which is completely novel now that we have different abilities with Generative AI that's beyond which we could have done with just pure discriminatives for other solutions. I'll also mention here that in the data platform side and that like the cloud architecture, you'll get new ideas, new concepts coming about, such as new database structures for effective databases that are becoming a bigger and bigger deal. So, things like Pinecone or Milvus that existed a couple of years ago still getting bigger traction because a lot of the stuff that we need to do needs to be optimized for the outputs of these AI models.

Data Sets

The next big component is data sets and data. Data science is completely founded upon data and the quality of it. Maybe two or three years ago, these were on the order of a million images or couple tens of millions of documents. Nowadays, we're up to the scale of billions of images. We have data sets like LAION that are 5 billion, that are scraped of images and pairs of text that go with it. And this is starting to get to the like, all available Google images that exist out there that are of high quality. So, we're starting to get to the confines of what is possible even with, able to get everything in from the internet.

On the tech side, these things are terabytes and multiple trillions of tokens and words that exist out there. And one of the key things to notice here is that the types of data that you use to train your models are going to be the limitations of your model. So, by that I mean if you don't train your model with any co-data in the background, so these things historically were only trained with pure text. And we actually cut out the code part of it. If you don't do that, if you don't have a code part, it's going to have a very hard time generating code because it has just never seen it before. And so, the models that we train are learning from the patterns from the data.

So, being aware and injecting domain information into it is going to be critical. So, the coding has been injected recently and a lot of times we're looking at just the common crawls or the common like web pages on the internet. that's been a little bit sanitized, but also things like books or peer-reviewed articles or journals so that we have some clean data set out there. And, these are, again, multiple trillions of words and it covers a large variety of the data that exists. But when you get to like things that are more niche, you might have to do something new. We'll talk about fine-tuning as we get there.

The other thing to mention is that as you construct these data sets and as you think about building these things and how they're put together, not only like the final thing that we're thinking about as important, but also like how we clean those. So, are biases in data sets removed? Are there issues with like gibberish out there? Like, do we want things like the fact that a lot of the text content out there is fictional? It means that you have a lot of fictional prose in your content that you can generate. But also, maybe that's not what you want to do because you want a more factual base.

So, those kinds of things really, at this scale, have huge implications for downstream and for applications.

Foundation models

Foundation models … obviously, things like GPT-3/-4, DALL-E, diffusion models, GANs, foundation models are becoming a bigger and bigger deal. And what I mean by that is over the last 10, 15, 20 years, we used to have very dedicated models to solve dedicated things.

So, you would have a model to detect cats vs. dogs, to detect the outline of cats vs. dogs separately to just a classifier. And these models were subdivided into very specific categories. Where you can imagine we're learning from the old physics days where you had a physics equation that defined one particular problem set and one particular way to solve it.

Same thing with language. We used to have a particular model to do entity extraction. So, being able to extract keywords and the one model to do sentiment analysis, one model to do classification like all these things were disparate models that existed, that were trained separately on different types of data and used differently. In the last two to three years, foundation models have been coming in bigger, bigger deal where we're training these foundational models in an agnostic manner so that they can do downstream tasks all simultaneously from the model.

So, an LLM these days is able to answer questions, do commonsense reasoning, extract things from text, translate things all without changing any of the underlying weights or structure of the model itself. So again, that's sort of the paradigm we're moving towards. Moving towards more of a generalizable model that will tackle more and more use cases. And from here you're thinking about LLMs that are just text or traditional CNNs, convolutional neural networks, or GANS to do images.

But we're also combining those modalities so that you'll have multimodal foundational models that are trained on both text and images. Text, images, and speech and everything combine into one model that can answer and do just arbitrary number of things that people need, limited by the types of things that we have data for. I'll mention one caveat here is that there is the foundation model, and the interpretation is really just models that can be used to do downstream things in a task-agnostic manner.

So, just one model for many, and that's not necessarily limited to just generative models. So, there are discriminative models like BERT is technically a discriminative model where it's still a foundational model because it can do multiple tasks in one foundation, one thing, but it's not generative. So, there is slight differentiation. The idea of generative models and foundation models coming together a lot of times is simply because to do things usefully and to do foundational models and have a task-agnostic manner like understanding the full distribution of data. Going back to understanding the distribution of data just naturally leans towards a more generative model.

Model training

Moving into the penultimate stage of model training. So, traditional deep learning methods basically have a pretrained model. Maybe it could be an LLM or a computer vision, feature extractor. And what happens is you put in a document and known outputs.

So, let's say I have legal documents and the known kind of legal documents that you want, Class B. And how hundreds, thousands, tens of thousands of examples of paired known examples from input and the

expected output. And in traditional deep learning, what happens is the model will learn these patterns of how to associate what's in the text or what's in the image to the class output that exists.

So, it learns patterns to the mapping between the patterns and the output. And then when a new job comes in, what happens is you just push the model through and I will basically say, this is the patterns in the document, what is the most similar class to predict this work. I'll give you some distribution, but they'll give you like a prediction of like it is class X.

Okay, so that's traditional. Foundational models are generally trained in the self-supervised manner, meaning that it's using the raw data in a way to learn the patterns in the entire data set without a firm division of what you're trying to do. So, instead of having the document and a class that you want to associate it to, you actually use the first half the document to predict the second half of the document.

So, in this case, you give the model inputs of, “Hey, diddle, diddle, the cat and the fiddle, the” and you train them all to predict the “cow jumped over the moon.” And so, in this manner you just scan over all the available trillions of texts that we have, and the model will learn the patterns to what is the next word likely to be right over the data set.

And so, by doing this, it's more like a fundamental understanding, intuitive nature of what it is for text to exist. And it starts to learn patterns in the data set of “the” and “cat” are followed by “Hey,” and "the fiddle …" And it seems a lot of these examples you can have like, oh cat is more noun-like because you replace cat with dog, for example here, and it still makes semantic sense, and you'll see examples of that.

And so, it starts to make these patterns and understanding and the representation of like what it means to be each of these entities. From there you can do an optimal fine-tuning step. So, as I mentioned, there are instances where your data set that's been pretrained on, doesn't have the exact types of documents that you want it to. And in this case, what you need to do is you can fine tune the foundation model by giving it something that's in the same domain.

So, it's just adding it to its training corpus. In this case, "This nondisclosure," and then separately trying to predict “agreement is entered on May 17.” So, by giving it maybe a couple thousand of examples, it will learn these types of nuances for your particular data set. You don't need that many. You have the core language understanding from the LLM, but now you have this domain expertise that exists.

So, here you might say that, oh, the "nondisclosure" and "agreement," "agreement" is more likely to be capitalized because it's my kind of data rather than just generic text scraped online.

And then the next step is generally these days, instruction tuning. So, the LLM is very good at predicting the next word and it’s trained across trillions of documents, transitive of tokens and words.

But about a year ago, InstructGPT came out and it came to the idea and came to conclusion that to get models to do things better, you would basically give it a more formal structure of what it's looking for. So, by that I mean you'd give it say, “What date was the end of April?” and the answer.

So again, it's going similar almost like we’re back in the standard deep learning era. But you do this across different types of reasoning, different types of questions, you can even do translation and like always had things, but it's really prompting it with the question and an answer. And so, the idea here is

that by injecting this sort of structure into it it expects question and then to give an answer is really the key thing here in terms of instruction tuning.

And the benefit that you get is that instead of trying to predict the next word, you're trying to predict like the next word, but more primed towards the fact that it's meant to be, you know, an answer to the question you're asking. Because you can totally imagine in the corpus of data that online, a lot of the questions like, “What date was the NDA put in effect?” is followed by another question.

It's just a list of questions that are like a template for what to include in the survey. And, and the language model by itself would just predict the next question—which, or what, or who. Whereas really you wanted to make sure that you're gearing up towards a real use case where you provide a question and instead of providing the most likely next word, predict the next word that's an answer, and that's the instruction part of it.

And then finally, at inference time, what you can have is any of those three things. So, foundational models, or fine-tuned models, or instruction fine-tuned model or even reinforcement learning with human feedback, which is the latest that we'll talk about in the next workshop. But all these things are basically in the middle here, and the input that you give it are instructions.

So, translate to French, “the dog” and the output of the model because you've seen this pattern before in data will give you, “le chien.” If you can also give it some examples, it typically does a little bit better. So, if you give it an example: Translate to French, “the cat and the shop,” and then give it: Translate to French, “the dog,” and then ask the model to predict the next two words.

It will likely match that pattern in the outputs. And so, the Generative AI models will basically generate the next words to give you that answer that you're looking for and to generate.

Applications

And finally looking at the application layer. So, you talk about from accelerate to create, we do have definitive use cases where we can do this from. Like coding is a prime example where we've done this before; is where you type in a prompt with like three or four bullet points and you generate a modeling like a downstream modeling, it’s just phenomenal. And in our experience when we're using it, it works maybe 80, 90%. And then you can just plug and play that code into the into the workplace.

And then you still need the human-in-the-loop right there because there's still a high risk. You can imagine, like a generative model doing remove dash R star and delete everything from your computer. So, you basically have to oversee it because you're unconstrained in the general part. But it does speed up a lot of work and it does speed up writing as one example.

Then there is automate. So, things that can be automated in a low-risk manner, like employee … taken off the hands of people. Again, this is a great idea. And like what we've done in the past for the pharma companies is plain language summary, summarizing complex medical documents and jargon, multiple pages worth, into a couple paragraphs of patient-friendly and patient-readable information.

That's such a really good use case of automating the workflow. The patient summary doesn't have to be completely correct, and if it does have to completely correct, maybe it goes into the “Accelerate” bucket. But generating a lot of this content is very painful, and you can automate a lot of that. Also, things like chatbots and those kinds of things that exist in “Personalize” are the next step.

So, I'm looking at the “Personalize” bucket here. We've done this in the financial space. It was where you get an agent and you're replying to it and making recommendations. And the agent itself that has been trained on this as a language model will understand how to answer these questions and reach the resources that it needs and being able to tell you the information you want and then remove a lot of the human call center that needs to be there.

And again, you can also personalize and tailor the responses to your needs. So, if you are more likely to want advice about one type of topic or the other, or the tone of the topics that you want it more curt, and just the information, again, it can be tailored directly for each person.

Then moving on to “Simulate.” This is one of our senior data scientists. Data science manager here at SFL, where we are. This project was used to basically simulate an avatar of each person. So, you can do things like break-dance with the person. So, in this case, what we did was, we took a video of 2D of someone spinning, basically a video from spinning, just rotating around.

And then from there, from that mapping, we can generate the avatar to do things that otherwise you can’t do, and you can analyze the behavior and all that kind of fun stuff.

And the last one is “Creating.” Creating things that otherwise could not have been done. And this is me on the image. I can’t grow a beard this thick unfortunately but being able to visualize and generate a marketing content for a facial product company or so on. This is something that wasn’t possible before, and now it is amazing that we can come see this.

Topic 3: Life Cycle of a Project

Generative AI Solution Life Cycle

And moving on to, I guess, the last section, the life cycle of the project. So, the first one on the left is being able to identify a need. So, as with all consulting, as with all use cases and business decisions, you need to fundamentally understand what you’re trying to solve and whether or not Generative AI is the solution you want to pursue.

Phase two is selecting the right algorithm, architecture, design, and start building the models that you need to use to do so.

The third is major components need to be evaluated and tested, and there’s a lot of nuance in how Generative models are tested, and I’ll talk about that are required to ensure your business outcomes are hit. And then finally, it’s the MLOps that’s typical where you’re interacting and viewing things live and in production and monitoring and validating and making sure that data flywheel and the usage flywheel of collecting issues and edge cases and get refiltered back into the workflow and improve your models as you go.

Use Case Identification

So, the first step is to identify your use case. Again, very similar to any consulting or any sort of decision that you want to make, trying to prioritize projects, it’s whether or not Generative AI is the right solution. And 90% of the time, there is an easier, simpler, or cheaper way to do it that might not require Generative AI. So, think about things where you can say like, cost is not really an object where you need to provision thousands of GPUs, tens of thousands of dollars to just support that. If that’s not an issue, maybe Generative AI is the solution. If you have the data that surrounds it. If the use case that we’re

talking about is something that is low risk, then maybe you can automate it. And if it's higher risk, then think about the workflows of how it's used by the end user. There's human involvement there. The data available again as you have ideas of what you need to build these models. You can have a ballpark estimate of, we need a couple hundred examples, couple thousand examples, maybe or discriminative models, something that's simpler.

Whereas if you want to train or fine-tune some models, you go into the thousands or tens of thousands. And then when you get into ... you want to train your pure language model from scratch, like domain expertise, then you're starting to get into the millions of samples that a lot of instances don't require this kind of thing. So, we'll talk more about this trade-off between data and the workflow in workshop three, just to give you some ballpark numbers of what to expect. And then obviously the ROI, how much is successful? A lot of these situations and a lot of these generative models will give you 60%-80% of the way there. But having some human-in-the-loop is that cost savings to you fundamentally.

And then just to highlight again, so generative models will cost more, in general. They might not be as useful in high-risk scenarios without having human oversight. And the output itself is more generative, more creative, which means that there are more risks involved in deploying these things. I'll talk more about that in three slides.

Model Selection and Development

The next step is thinking about, let's say we have a use case now. Now what? Which models do we use? What's out there? How do we even, like adjudicate good and bad? So, how would I think about it in terms of data modality first. So, let's say you have text or you have images, are the two typical common modalities we have. You can have speech, you can have signal processing or time series. You can have audio, you can have videos. But think about what type of data that you're getting in, and then you think about what exists out there. Right now, the biggest ones and the best ones are in language modeling. So, anything with text related. By text I mean, text, and code, and images. Images are generally used in diffusion models, GANs, and VAEs to some extent, but a lot of the time those two modalities are pretty robust and trying to get more and more mainstream, as you guys are seeing. But those … outside of those, they're starting to get more and more niche of whether or not you even use, or whether or not a Generative AI solution even makes sense right now because the underlying foundational model, the underlying data just doesn't exist. So, then you got to think about the accuracy and the speed that you need. A lot of times you can sacrifice a little of accuracy if you get the speed weighted. So, for example, with say something like autocorrect, I want autocorrect work very quickly, and it's okay that it's 50-50 of whether it's right or not because it's very low weight and I can just ignore the output. A lot of times in Generative AI workflows that accuracy and speed is a bigger consideration you need to think about because you're never going to get the accuracy of 100% for generative model just by nature of it being creative and sampling from this blob of distribution of data. So, it's not like how much you can tolerate for accuracy versus the time it takes to run these things, because these are big models, and these require huge GPUs. And then you think about the development and the budget and the budget for deployments, like how much you actually want to charge people, or use the service, or is it 50 cents, 30 cents? Thinking about how much ChatGPT is used these days. So, they're at the level of maybe 20 or 30 cents per question that gets asked and that needs to be answered. And if you have tens of thousands of these things, it starts to get very, very expensive to just run and to figure out. So, maybe a lighter weight model, maybe less performance, but can run faster and with less compute. So again, it's very

complicated to think about, but I could think about in a hierarchy, does a solution, a prepackaged solution exist? Or someone that's worked on it had robust modeling behind it. And then thinking about the accuracy, speed requirements and the budgets involved.

Metrics, Scores, and Benchmarks

Then moving into metrics, scores, and benchmarks. So, this is a huge space where there's going to be a much more intricate need for understanding model-specific metrics. So, what I mean by that is, let's say you generate an image of a cat or a dog, or generate the summary of text or generate some code. How good is that? How good is that image of a cat? And it's very hard to tell, right? Like there's no yes/no, this is class A/class B kind of accuracy. There is no rule out of the gate. Here is the thing to use. It's more subjective. And that subjectivity is what kills a lot of value or the measurement of value. And if you can't measure the value, like how do you iterate on the model, how do you iterate on the process? Like is it even doing better than what you did before? So, model-specific metrics are key in differentiating, something you need to think about in Generative AI. Business metrics are the things you normally think about, like click-through rate or conversion, or the number of edits you accept from the recommended. Things that actually have definitive dollars and cents behind them that you can track. The issue and reason why you need both of these things is that there might be a whole slew of reasons between generating an image and why someone bought that or not. The noise from the instance of generation, it might not be the generation is bad. It just might be that that's not that person's preference, for example. And being able to get as close to the model output and evaluating that will give you a much tighter way to manage and make more systematic way to develop as you go.

And just a couple examples. So, in images, one of the typical metrics to use is this FID score. So, the FID score is the Frechet Inception Distance. So, first of all, an inception score is how much like the generated image is compared to other generated images of that class. So, let's say I try and generate a person. How much that person looks like according to a model. So, we have really good models that can detect, you know, cat vs. dog, dog vs. person, person vs. bike, those kinds of things. So, we use that model as a proxy and basically say that, "Is this image very person-like?" and if it's not, then there's an issue. And so again, let's just think about this. If the image is very person-like, then it gets a low score FID score or low inception score. Saying that, yes, it's person-like and the ideal score is actually one step beyond that.

So, let's say I can generate really good images of cats or people. And my metric is how people appear. The next step is, okay, I can generate really good examples of one person. The most person-like image that I can generate. I can generate one image of cat, and that's the most cat-like image that I can generate. The next step is understanding the distribution of data. So, you don't want just one type of cat or one type of picture or person; you want a distribution of them. And so, the FID score is really trying to look at who generates these images and look at the distribution of the images and compare it to the distribution of real images and making sure that the color content and the spacing, and the size of people are different. And so, you can cover a lot more of the space.

And on the language side, you can have things like perplexity. So, perplexity is just a fancy way of saying how likely a sequence of words are to exist together. So, "The cat sat on the mat," for example, is a very likely sequence of work because that's existed, tens of thousands of times in your training set. Whereas "The cat sat on the toaster oven" is very jarring because it doesn't exist in your training set. So, that might only exist once, definitely not twice with that same cat. But you think about how often that exists and like just probabilistically, and then you can basically weight things of like this is a likely sequence of

words that exists. So, the more likely a sequence of words is to exist, the better we're saying that the language model was generating text. It just sounds more realistic. And then downstream you can do things like accuracy. You can actually have metrics on question-answering systems, or how well it does and classification. Again, leveraging a lot of the aspects of foundational models being able to just do this out of the gate. But really those are more task specific. And if you want to just put raw performance of your LLM for your use cases, you can use the perplexity as a good proxy.

And then finally the data flywheel and continuous improvements. So, this exists again and in almost all AI solutions. You have a model that you train up, you have your data and you put it in, and then you deploy it. You get that feedback loop going. All the users come in and say, "That doesn't seem right." And you collect that data, you put that into your model again so that you update it for those "edge cases" and then you deploy it again. So, that flywheel happens again and again, is continuously overtime. The main difference here with Generative AI is that this flywheel will need to be super tight. And what happens generally in development is you do that personally. So, I try a version of my model first and then I try the output to make sure that makes sense. I mark down everywhere that doesn't, I make modifications, and I do the same thing again. I will redo the model again and again. When it gets good enough for myself, I present it to my team. My team will basically go over the same thing, and try and squeeze interesting things out of it. And then when it gets to a point where we can't seem to find an edge case, then we can deploy it to a broader pilot study. I think that sort of iteration is going to become more and more quick-paced because of the need for control in the Generative AI models. So historically, when you have discriminative models, it's only ever going to be a cat or a dog. And maybe if the model is not as good, there'll be more. Oh, there's an 80% accuracy of like those cats or those dogs. Whereas the generative model, because it's more creative and it's pulling from the distribution, you can have weird situations where what's generated is completely nonsensical. Like you're asking for "Is it a cat?" and "Is it a dog?" And the answer is, "It's a penguin" or something more biased or more toxic. You have no way to control that unless you put a lot of constraints around it. And every time you have a new constraint that you put around it, someone's going to find a better way to work around it. And it's just a matter of how, like the raw nature of having create a model and having generative models because you can't control the output. And if you did, you'd probably use a discriminative model anyway. So again, because of that output and because of that unconstrained nature, you really do need that flywheel to be tighter and tighter and it's going to be orders of hours they need to respond to it rather than days or weeks that just aren't being able to do so.

Project Planning

Now that we have discussed the major components of a project life cycle, I just wanted to, at a high level, explore the aspects of project planning and specifically the roles that are existing within these Generative AI solutions.

A lot of these are very similar to any AI use case that we are starting to solve. So, the PM, for example, having a firm understanding of the business case, firm understanding of domain specifics, and then also understanding of how decisions are made in that communication role of managing the risks and the outcomes. Again, very heavily important in Generative AI. Data scientists have more tools nowadays with being sure that they understand the concepts and those kinds of things and being able to develop machine learning solutions. The engineers will come at bigger and bigger field, and specifically in Generative AI as the workflows get larger and larger with the size of the data and the size of the models.

And there's typically machine learning engineers, I suppose on this is where we're starting to optimize models, trying to figure out how to optimize CUDA kernels and getting into the nitty-gritty of distilling models, or productionizing models for exactly what you want downstream. And this is going to become a bigger and bigger role simply from the cost perspective of saving fractions of a percent of cost will have huge implications for the ROI.

The machine learning operations engineer, again, it's going to be another, bigger and bigger role in Generative AI where you're starting to think about these solutions in these flywheels in the matter of hours and days rather than weeks and months. So again, a bigger focus in Generative AI solutions.

And then the software engineer and the subject matter expert, again, just very common roles within the EAI workflows.

And then finally I'll mention before I move on, the prompt engineer. So, this role has been floated as a necessity for interacting with, say, language models. And I think that the ability to generate and build good prompts will eventually sort of trickle down to the exact person that handles the that part of, that component of the solution. So, for example, if it's a report writing or if it's to do with project planning or mitigation of what potential risks can be involved, then the project manager will be using these things to just generate examples, generate text, generate content for you. The data scientists and engineers will be using it to generate code. And again, they'll have to get more and more familiar with how to generate that code and how to prompt it to give you that good, useful output. And then all the way down, basically the same sort of idea, and the ability to generate good prompts will be subsumed, in my opinion, by the roles that have to oversee that, that part of the analysis, that part of workflow. There may be instances where large organizations have dedicated people who are subject matter experts at building prompts. Again, this is thinking about doing this and reproducing this, you know, dozens, hundreds of times. Then maybe it makes sense to have some of it really understand the intricacies of how the model is trained and what it kind of expects in the prompt, because that's really what it is. It's the ability to mimic what the model is expected to give you, the output that it expects to give you. And that role really is just an intuition of understanding how it's trained, instruction-tuned or what have you, and then figuring out the wording around it. And some of these things also will be abstracted to platforms like LangChain, where these prompts will be systematically created for you. And over time, the models will get more and more agnostic to the exact wording of things. And it's really not a limitation of models these days than anything else.

Project Planning

And finally, in the section I want to talk about project planning, and a lot of these are again, very, very similar to what happens in any AI solution that we build.

So, we are "exploring deeply," being able to understand and comprehend what's in the data and how to clean it up. And if there are any issues in the database, there are missing values that shouldn't be there, or if there are correlated features and problems systematically in the data set that we need to address before you can build a solution. So, exploring deeply for these kinds of situations is going to be a bigger thing because we have more data to analyze and the output also is less constrained.

"Iterating quickly" is again just another general AI workflow that we adhere to. Being able to try three or four different models, try three or four different approaches very quickly so that you can figure out which one is the best, and then tackle that one as we go. And it's really just to reduce the risks that is

involved. And a lot of these things are generative and a lot of the data science part is the science of understanding how these things work. And there is a lot of trial and error systematically trying to improve over time, but really being able to do that experimentation is key to any sort of data-driven solution.

And then finally on the left is “educating widely.” Again, as these tools are more and more complex, getting stakeholders to understand the risks involved and getting SMEs to understand what’s possible and what might come about is a huge thing for specifially Generative AI, but AI in the whole field.

And more specifically about Generative AI is “provisioning early” and getting the compute and the resources necessary, like even trying to get the number of GPUs in a distributed system requires some time and a lot of cost. And the idea of being able to plan very thoroughly on this part will mitigate the risks as you go. And you’ll find that you kind of buy AWS instances with these GPUs sometimes because they’re all being used by someone else. So, being able to think about it in terms of like weeks and months ahead will really be useful for probable solutions.

Then, “plan adaptively.” So, again with the Generative AI, and more of the creativity that these models produce, a lot of edge cases will come about and things that you don’t think about even more so than normal AI projects. But it means that you need to have a flexible mindset in how we tackle these problems or how we pivot from one thing to the next. It’s not that it’s not targeted and systematic. It’s that we need to plan into our projects and ability to be flexible and to adapt to these situations.

And then lastly is the “measuring carefully,” and that’s really towards the comments we talk about the metrics and the whole situation with generating metrics that makes sense to measure the quality of the generative output.

Summary

In conclusion, we covered some of the history and the components of Generative AI and also the life cycle of a project, and how it fits into the workflow that you guys will have to be involved in if you want to create a solution in this space.

Deloitte.