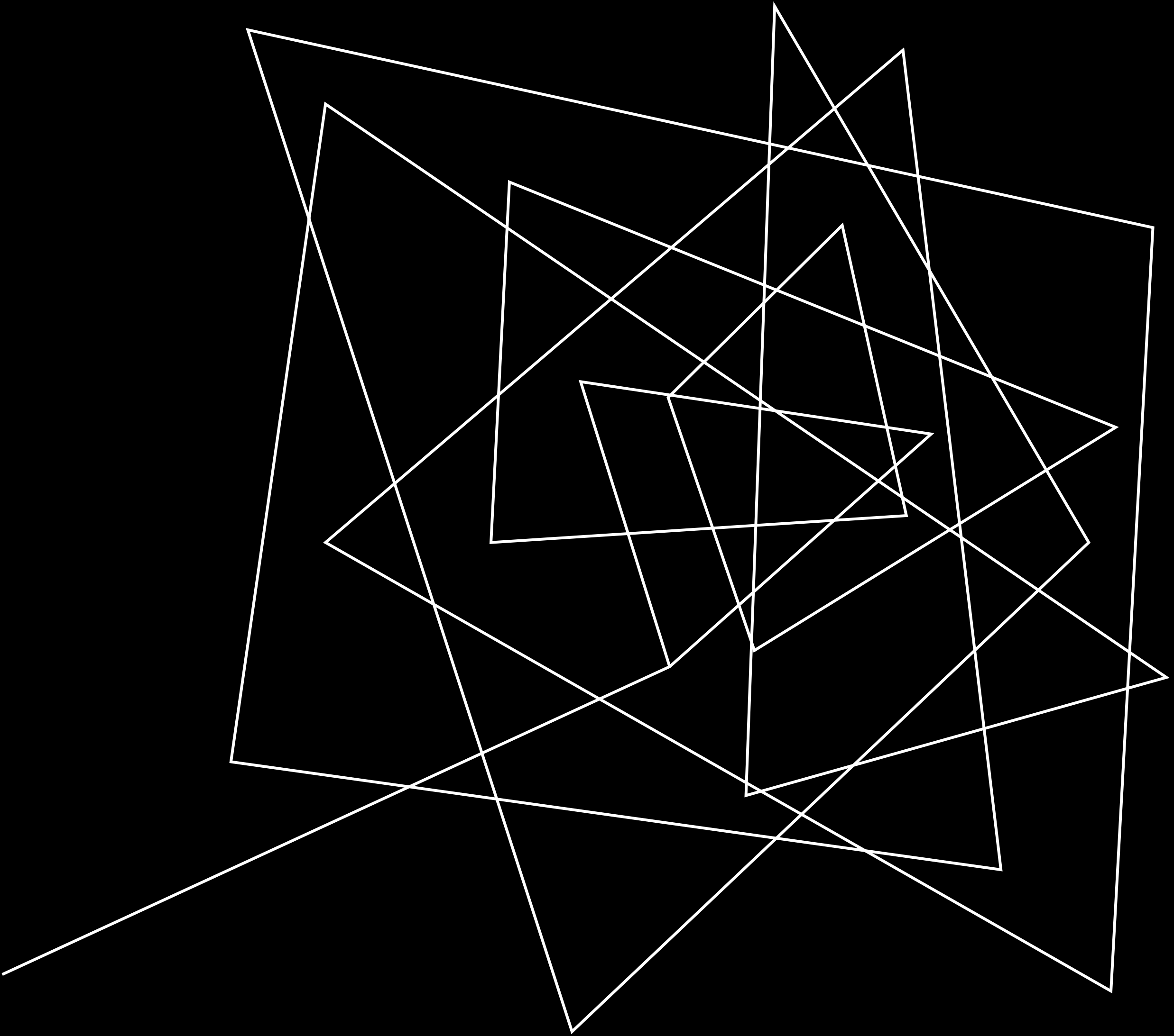
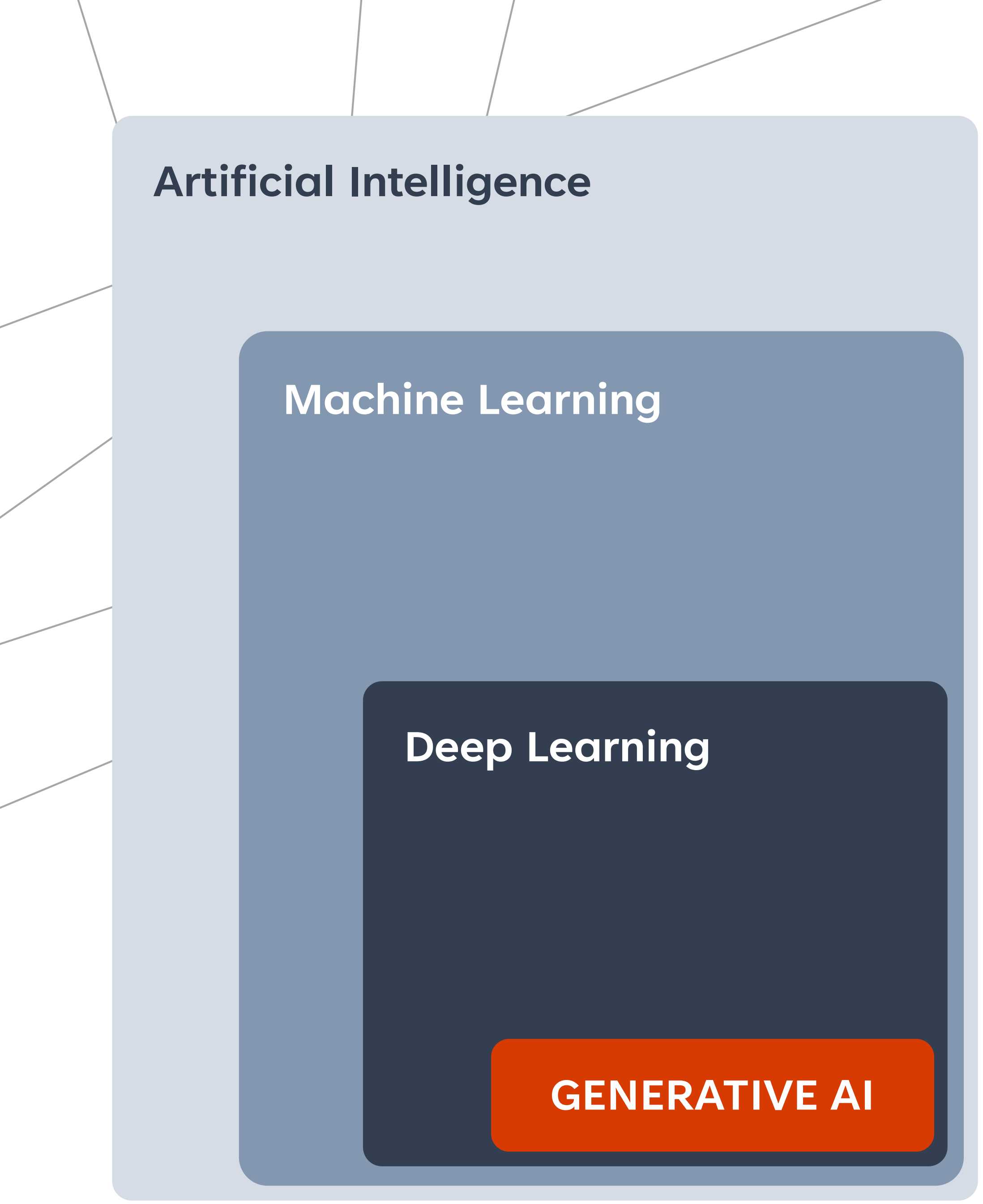
AZURE OPENAI SERVICES

Overview



AZURE OPEN AI SERVICE



1956

Artificial Intelligence

the field of computer science that seeks to create intelligent machines that can replicate or exceed human intelligence

1970s

Machine Learning

subset of AI that enables machines to learn from existing data and improve upon that data to make decisions or predictions

2010s

Deep Learning

a machine learning technique in which layers of neural networks are used to process data and make decisions

2020s

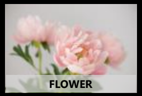
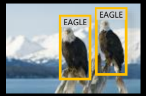
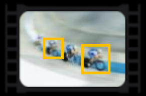
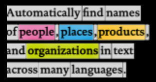
Generative AI

Create new written, visual, and auditory content given prompts or existing data.

AZURE OPEN AI SERVICE

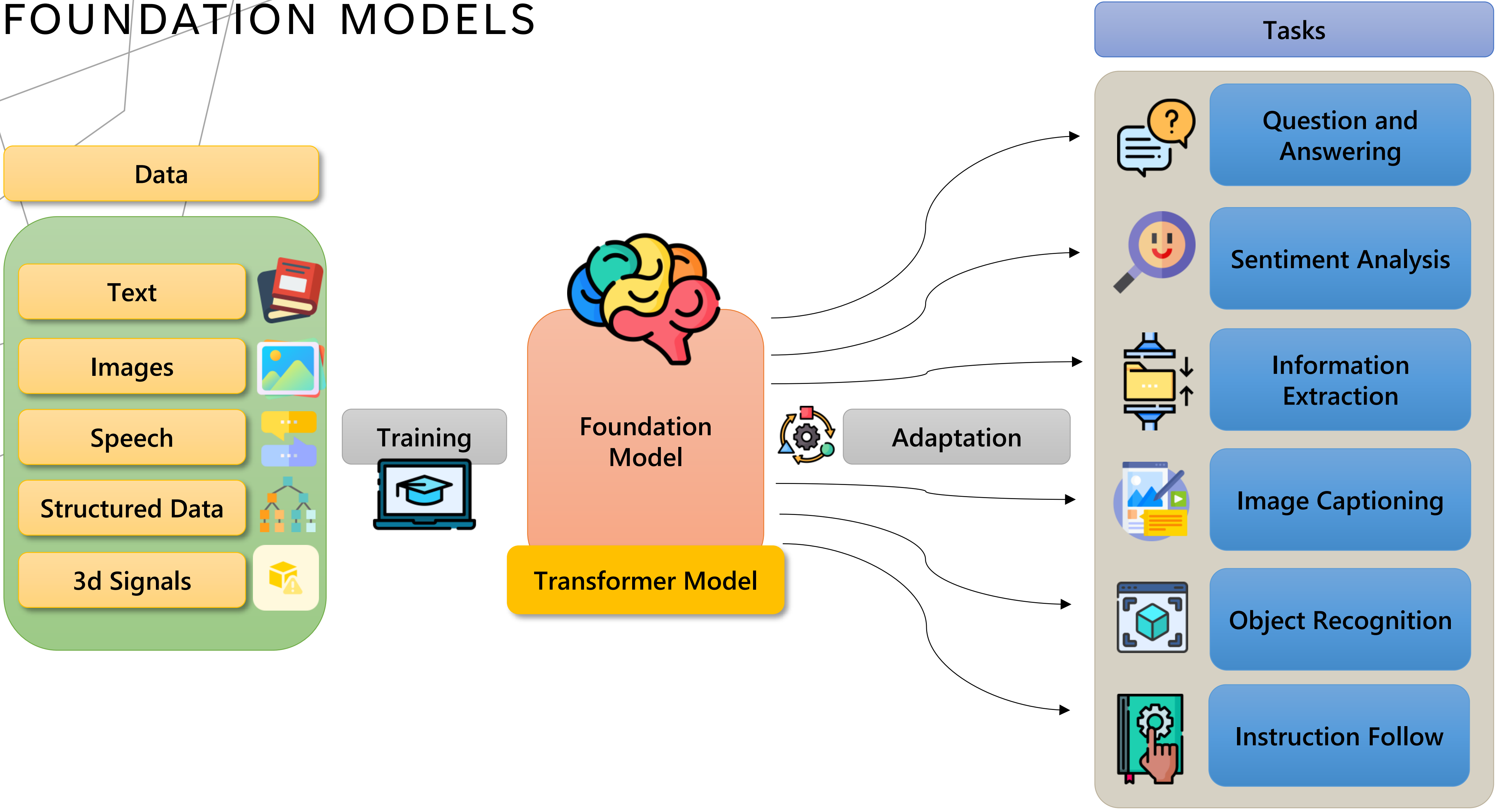
TRADITIONAL MODEL DEVELOPMENT

High cost and slow deployment—each service is trained disjointly

DEPLOYMENTS	Tagging Services		Spatial Analysis Services		Accessibility Services	Spatial Presenter	Azure Search, Video Indexer
TASKS	 Classification	 Object Detection	 Object Tracking	Action Recognition	 Automatically find names of people, places, products, and organizations in text across many languages. Entities	Topics	Sentiments
INDIVIDUAL MODEL (DISJOINTLY)	Classification Model	Detection Model	Tracking Model	Action Model	Entity Recognition	Topic Classification	Sentiment Analysis
TRAINING DATA (w/ ANNOTATION)	Tagging data	Detection data	Tracking data	Action data	Entity data	Topic data	Sentiment data

AZURE OPEN AI SERVICE

FOUNDATION MODELS



AZURE OPEN AI SERVICE

OPEN AI / MICROSOFT PARTNERSHIP

OpenAI

Ensure that artificial general intelligence (AGI) benefits humanity

+

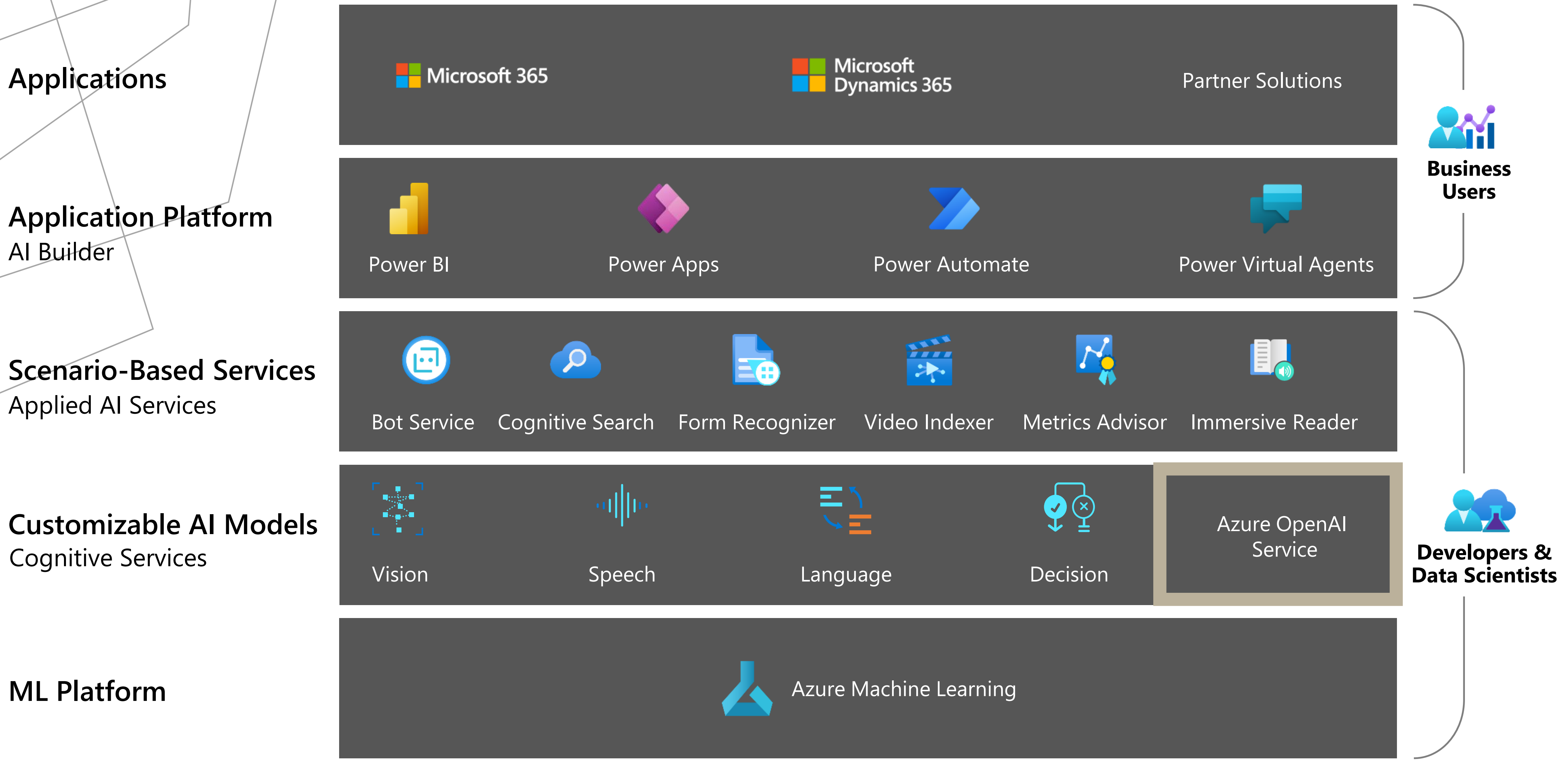
Microsoft

Empower every person and organization on the planet to achieve more

GPT-3.5 and GPT-4	ChatGPT	Codex	DALL·E 2
Text	Conversation	Code	Images

AZURE OPEN AI SERVICE

AZURE AI SERVICES



AZURE OPEN AI SERVICE

Azure OpenAI Service

GPT-3	GPT-4
DALL·E	ChatGPT

- Deployed in your Azure subscription, secured by you, and tied to your datasets and applications
- Large, pretrained AI models to unlock new scenarios
- AI models, some custom-tunable with your data and hyperparameters
- Built-in responsible AI to detect and mitigate harmful use
- Enterprise-grade security with role-based access control (RBAC) and private networks

AZURE OPEN AI SERVICE

MODEL FAMILIES

GPT-3.5

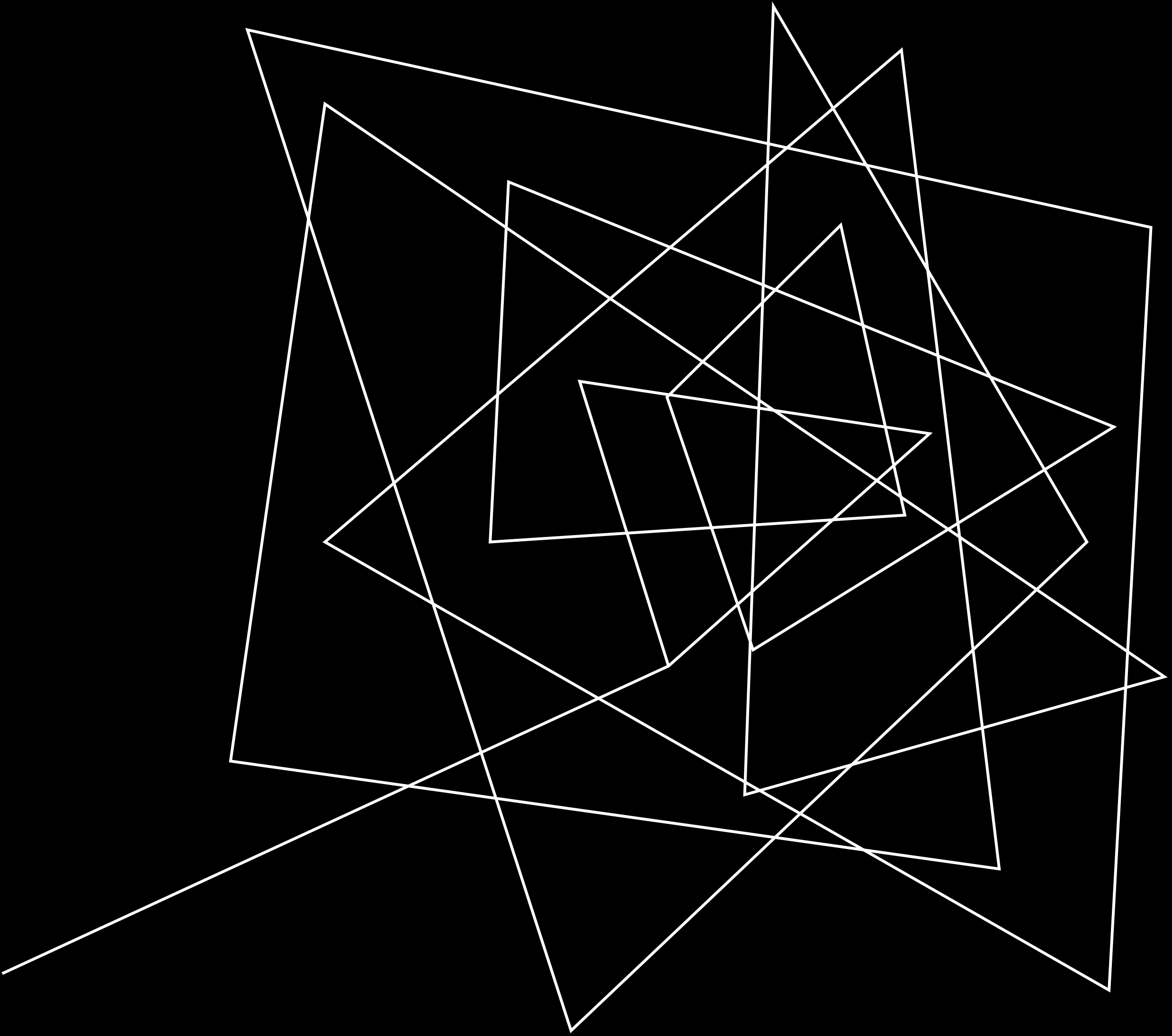
- Use-case specific models to optimize inference time and performance
- Suitable for a large range of use cases

ChatGPT (preview)

- Should be first choice for most use cases
- Most economical GPT model in Azure OpenAI Service
- For all workloads, not just Chat

GPT-4 (preview)

- Improved problem solving and reasoning capabilities
- Iterative refinement:
  - Paste in code errors & GPT-4 will fix for you
  - Iterate on stories
- Increased token limit - works well for long content (8K and 32K tokens)



CONCEPTS

CONCEPTS

TOKEN

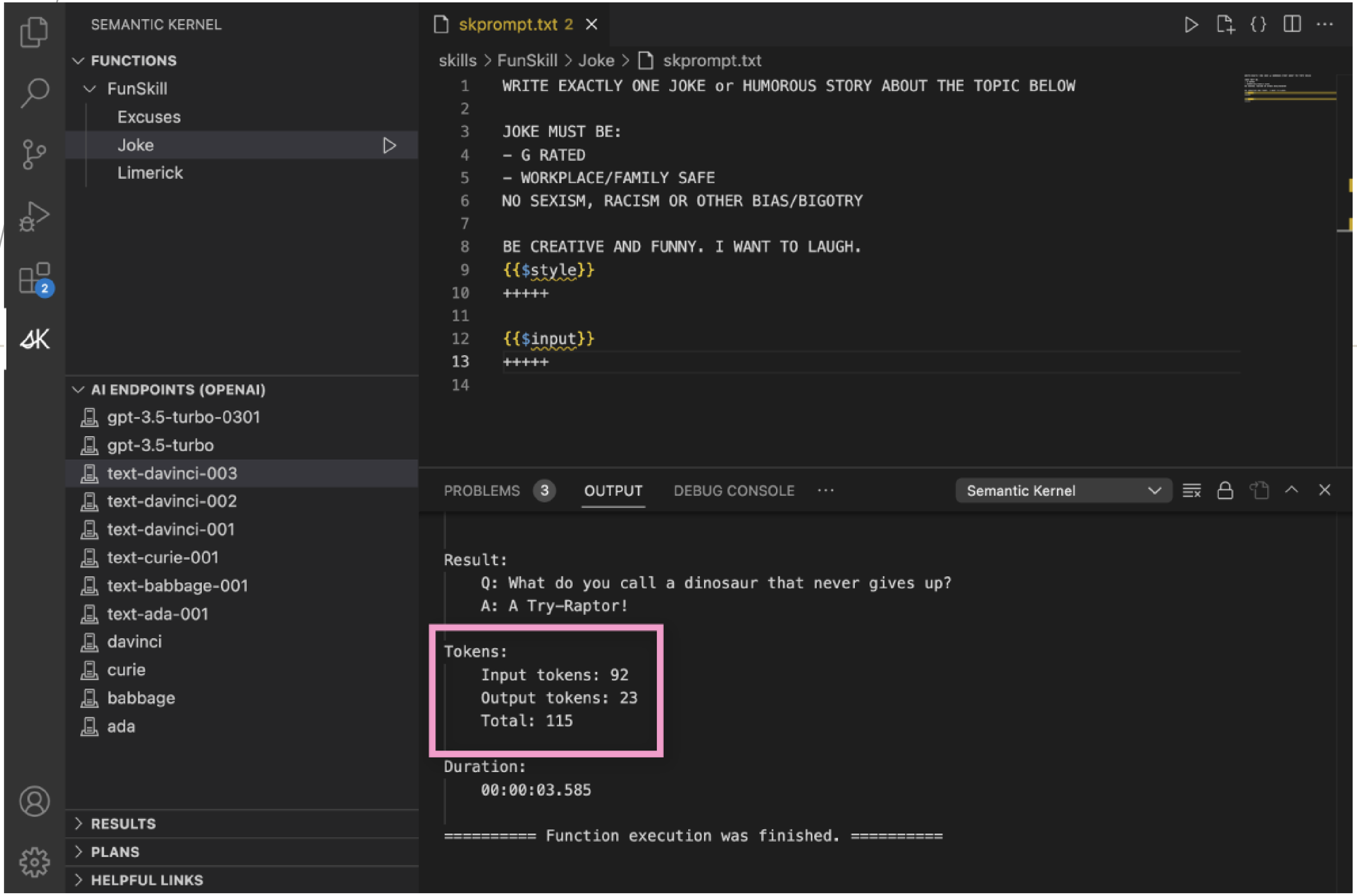
You can think of tokens as **pieces of words** used for natural language processing. For English text, 1 token is approximately 4 characters or 0.75 words.

As a point of reference, the collected works of Shakespeare are about 900,000 words or 1.2M tokens.

https://learn.microsoft.com/en-us/semantic-kernel/prompt-engineering/tokens

CONCEPTS

TOKEN



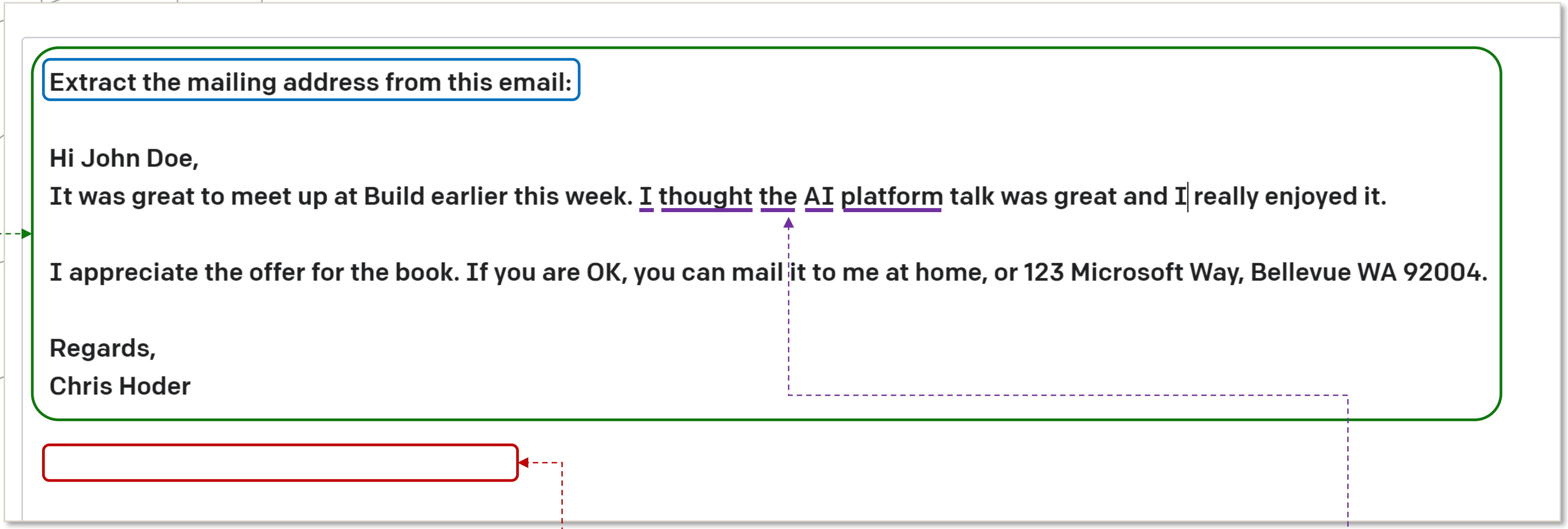
```
WRITE EXACTLY ONE JOKE or HUMOROUS STORY ABOUT THE TOPIC BELOW

JOKE MUST BE:
- G RATED
- WORKPLACE/FAMILY SAFE
NO SEXISM, RACISM OR OTHER BIAS/BIGOTRY

BE CREATIVE AND FUNNY. I WANT TO LAUGH.
{{$style}}
+++++

{{$input}}
+++++
```

```
Result:
    Q: What do you call a dinosaur that never gives up?
    A: A Try-Raptor!

Tokens:
    Input tokens: 92
    Output tokens: 23
    Total: 115

Duration:
    00:00:03.585

========== Function execution was finished. ==========
```

https://learn.microsoft.com/en-us/semantic-kernel/vs-code-tools/

CONCEPTS

PROMPT & COMPLETION


Extract the mailing address from this email:

Hi John Doe,
It was great to meet up at Build earlier this week. I thought the AI platform talk was great and I really enjoyed it.

I appreciate the offer for the book. If you are OK, you can mail it to me at home, or 123 Microsoft Way, Bellevue WA 92004.

Regards,
Chris Hoder

Prompt—Text input that provides some context to the engine on what is expecting.

Completion—Output that GPT-3 generates based on the prompt.

Token — partial or full words processed and produced by the GPT models

CONCEPTS

EMBEDDINGS

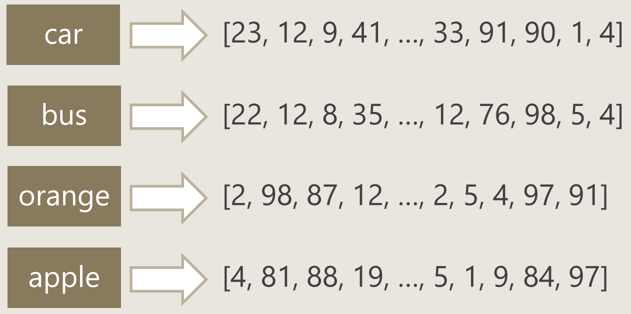
An embedding is a special format of **data representation** that can be easily utilized by machine learning models and algorithms.

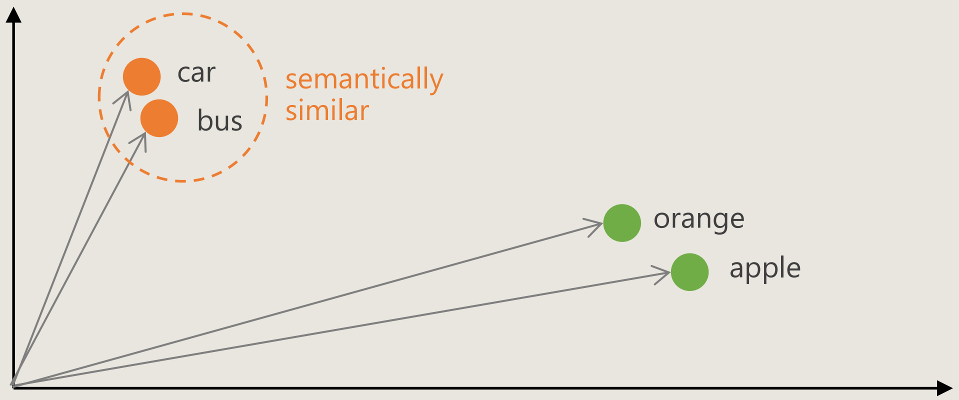
The embedding is an information dense representation of the **semantic meaning** of a piece of text.

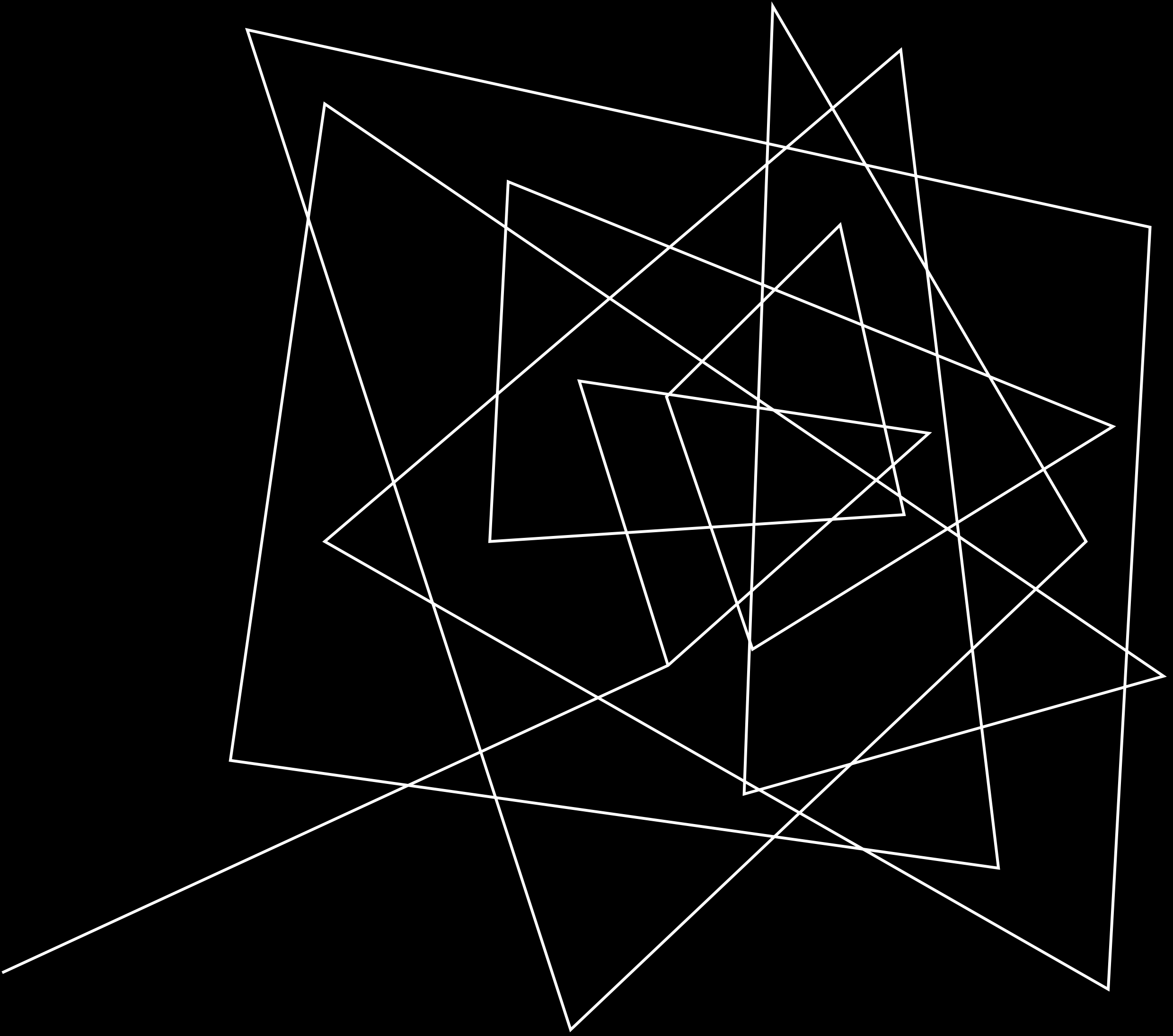
Each embedding is a **vector of floating-point numbers**, such that the distance between two embeddings in the vector space is correlated with semantic similarity between two inputs in the original format.

For example, if two texts are similar, then their vector representations should also be similar.

https://learn.microsoft.com/en-us/semantic-kernel/memories/embeddings





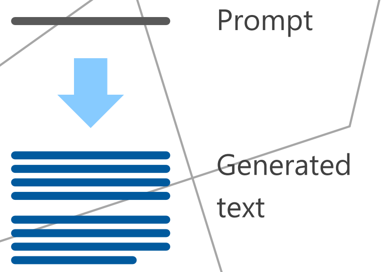


CAPABILITES & LIMITATIONS

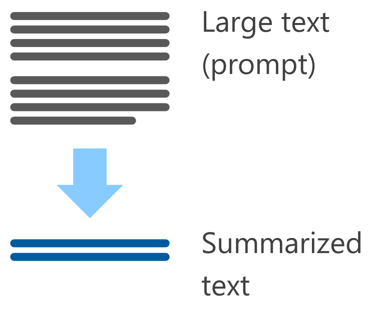
CAPABILITIES & LIMITATIONS

USE CASES

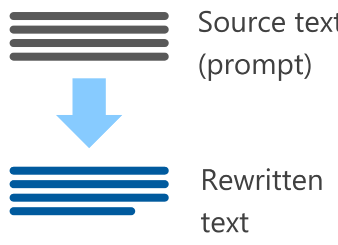
Generation



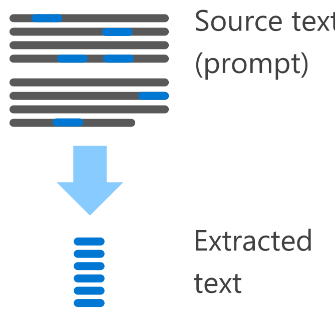
Summarization



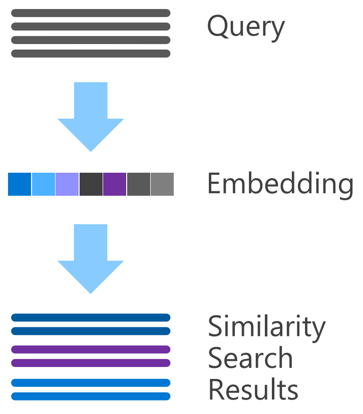
Rewrite



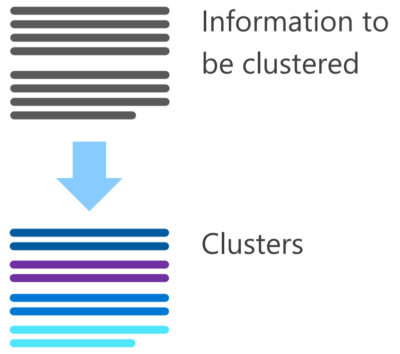
Extraction



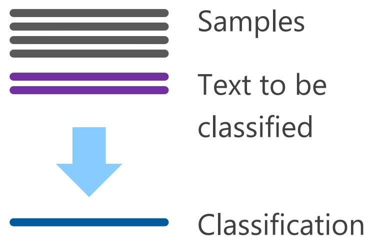
Search



Clustering



Classification

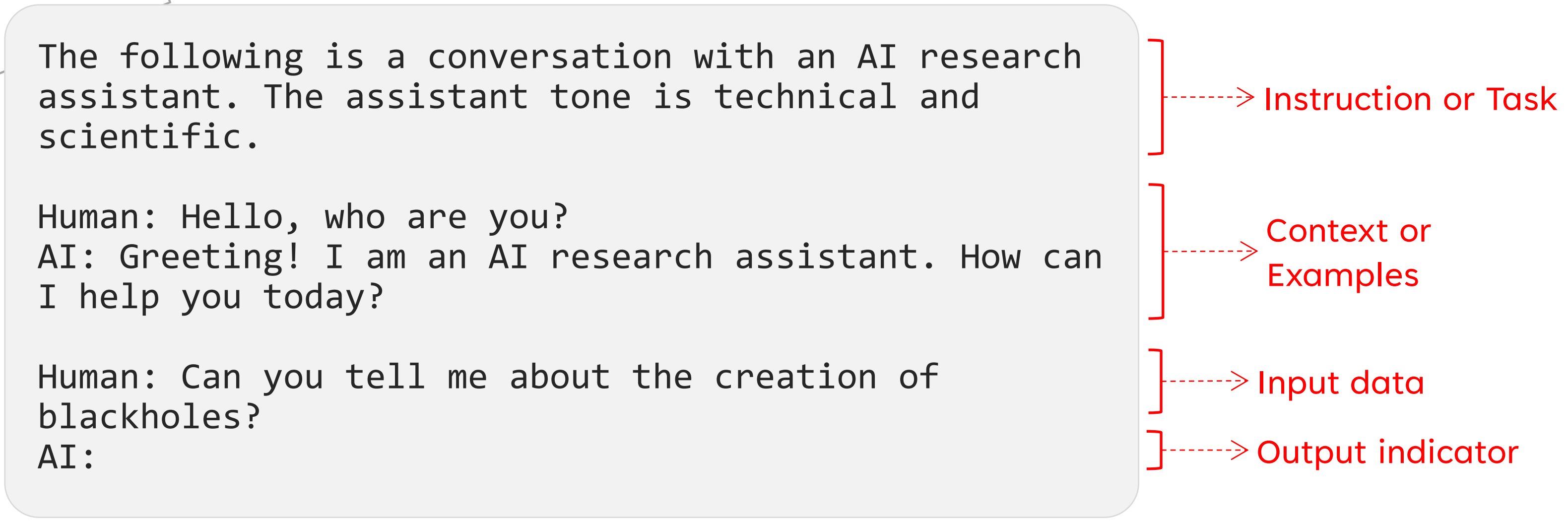


CAPABILITIES & LIMITATIONS

PROMPT ENGINEERING

- Creating the optimal prompt to elicit a particular type of response or steer the model’s output toward a desired direction.
- Essential skill when using LLMs. It is how you “program” an LLM.

Sample Prompt



```
The following is a conversation with an AI research
assistant. The assistant tone is technical and
scientific.

Human: Hello, who are you?
AI: Greeting! I am an AI research assistant. How can
I help you today?

Human: Can you tell me about the creation of
blackholes?
AI:
```

Zero-shot learning

```
Translate English to French:
Cheese =>
```

One-shot learning

```
Translate English to French:
Sea otter => loutre de mer
Cheese =>
```

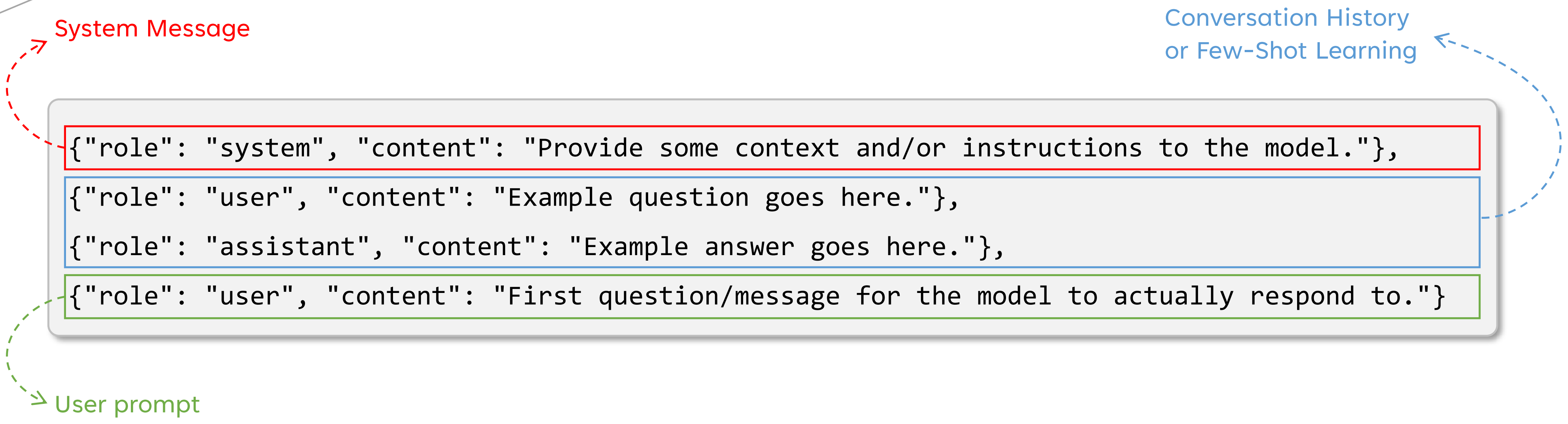
Few-shot learning

```
Translate English to French:
Sea otter => loutre de mer
Peppermint => menthe poivre
Plush giraffe => giraffe peluche
Cheese =>
```

CAPABILITIES & LIMITATIONS

CHAT COMPLETIONS

- Unlike previous GPT-3 models, the ChatGPT and GPT-4 models are specifically **designed to be conversational interfaces**.
- The conversational nature of these models makes it easier to interact with and to take advantage of the full power of its capabilities. This is part of the reason the model became so successful.
- The prompts used with the ChatGPT and GPT-4 models are also different than previous models.



```
{"role": "system", "content": "Provide some context and/or instructions to the model."},
{"role": "user", "content": "Example question goes here."},
{"role": "assistant", "content": "Example answer goes here."},
{"role": "user", "content": "First question/message for the model to actually respond to."}
```

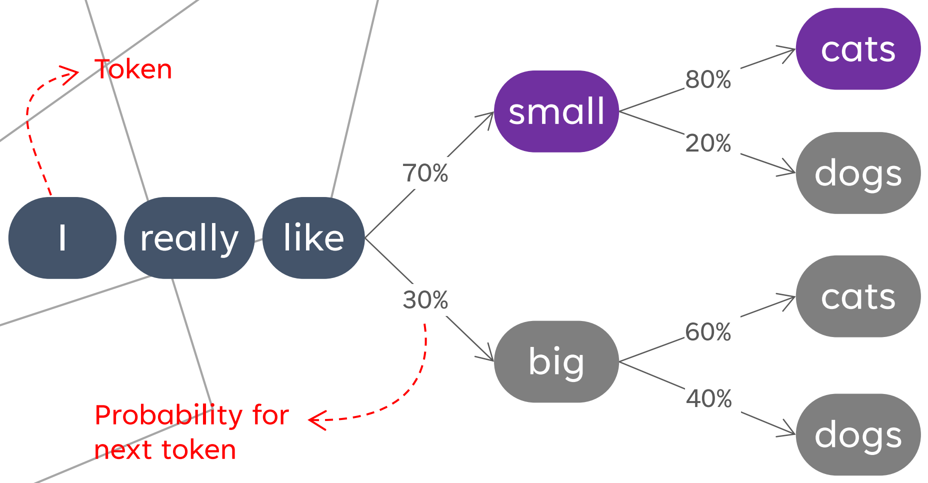
CAPABILITIES & LIMITATIONS

FINE TUNING

- Fine-tuning is a way of utilizing **transfer learning**. Specifically, fine-tuning is a process that takes a model that has already been trained and tune it using a labeled dataset for a specific task.
- Fine-tuning results in a **new model** being generated with updated weights and biases.
  This contrasts with few-shot learning in which model weights and biases are not updated.
- To fine-tune a model, you'll need a set of **training examples** that each consist of a single input ("prompt") and its associated output ("completion").

CAPABILITIES & LIMITATIONS

COMMON ISSUES

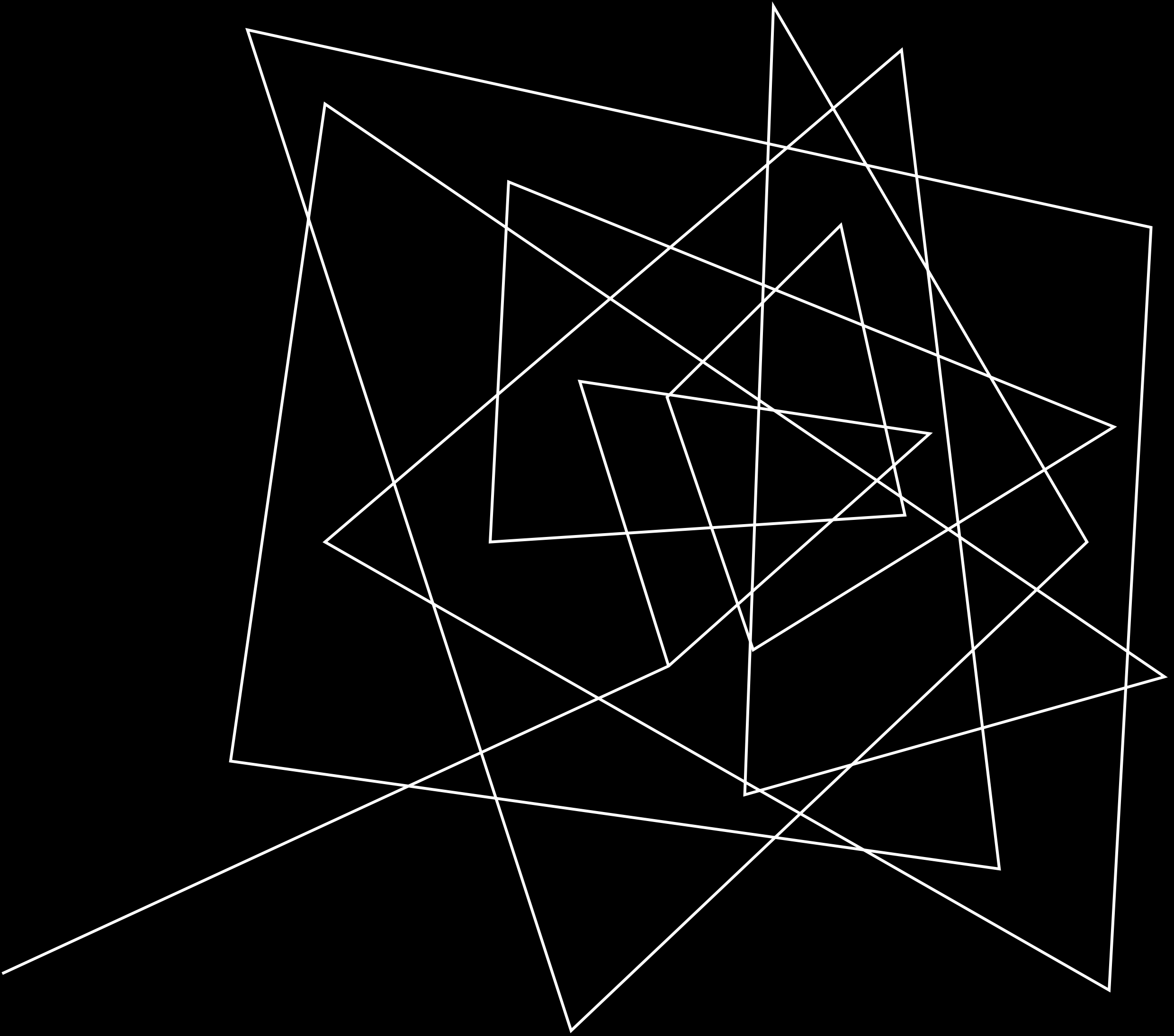


- May generate outputs that are not supported by the input or are factually incorrect. Also known as **hallucination** (or making up information).
- May generate outputs that are inappropriate, **offensive**, **biased** or harmful, and it may be **difficult to monitor** or correct their behavior.
- May become **outdated** or irrelevant over time as new information emerges. They can be retrained, but training requires a lot of computational power and time.
- **Limited token count** means prompts hold limited context which, when in a longer conversation, causes the model to "forget" what was said before.

CAPABILITIES & LIMITATIONS

REDUCING HALLUCINATION

Include	Restrict	Add	Repeat	Position
Include instructions to request the model not to make up stuff but stay with facts.	Restrict the output (e.g., choose from a confined list instead of generating free form strings)	Add Chain of Thought style of instruction, "Solve the problem step by step."	Repeat most important instructions in the prompt a couple of times.	Position most important instructions in the last making use of latency effect.



USE AI RESPONSIBLY

THE SCHILLACE LAW’S

USE IA RESPONSIBLY

THE SCHILLACE LAW'S

1 - **Don't write code if the model can do it;** the model will get better, but the code won't. The overall goal of the system is to build very high leverage programs using the LLM's capacity to plan and understand intent. It's very easy to slide back into a more imperative mode of thinking and write code for aspects of a program. Resist this temptation – to the degree that you can get the model to do something reliably now, it will be that much better and more robust as the model develops.

2 - **Trade leverage for precision; use interaction to mitigate**. Related to the above, the right mindset when coding with an LLM is not "let's see what we can get the dancing bear to do," it's to get as much leverage from the system as possible. For example, it's possible to build very general patterns, like "build a report from a database" or "teach a year of a subject" that can be parameterized with plain text prompts to produce enormously valuable and differentiated results easily.

USE IA RESPONSIBLY

THE SCHILLACE LAW'S

3 - **Code is for syntax and process;** models are for semantics and intent. There are lots of different ways to say this, but fundamentally, the models are stronger when they are being asked to reason about meaning and goals, and weaker when they are being asked to perform specific calculations and processes. For example, it's easy for advanced models to write code to solve a sudoku generally, but hard for them to solve a sudoku themselves. Each kind of code has different strengths and it's important to use the right kind of code for the right kind of problem. The boundaries between syntax and semantics are the hard parts of these programs.

4 - **The system will be as brittle as its most brittle part.** This goes for either kind of code. Because we are striving for flexibility and high leverage, it's important to not hard code anything unnecessarily. Put as much reasoning and flexibility into the prompts and use imperative code minimally to enable the LLM.

USE IA RESPONSIBLY

THE SCHILLACE LAW'S

5 - **Ask Smart to Get Smart.** Emerging LLM AI models are incredibly capable and "well educated" but they lacks context and initiative. If you ask them a simple or open-ended question, you will get a simple or generic answer back. If you want more detail and refinement, the question has to be more intelligent. This is an echo of "Garbage in, Garbage out" for the AI age.

6 - **Uncertainty is an exception throw.** Because we are trading precision for leverage, we need to lean on interaction with the user when the model is uncertain about intent. Thus, when we have a nested set of prompts in a program, and one of them is uncertain in its result ("One possible way...") the correct thing to do is the equivalent of an "exception throw" - propagate that uncertainty up the stack until a level that can either clarify or interact with the user.

7 - **Text is the universal wire protocol.** Since the LLMs are adept at parsing natural language and intent as well as semantics, text is a natural format for passing instructions between prompts, modules and LLM based services. Natural language is less precise for some uses, and it is possible to use structured language like XML sparingly, but generally speaking, passing natural language between prompts works very well, and is less fragile than more structured language for most uses. Over time, as these model-based programs proliferate, this is a natural "future proofing" that will make disparate prompts able to understand each other, the same way humans do.

USE IA RESPONSIBLY

THE SCHILLACE LAW’S

8 - **Hard for you is hard for the model**. One common pattern when giving the model a challenging task is that it needs to "reason out loud." This is fun to watch and very interesting, but it's problematic when using a prompt as part of a program, where all that is needed is the result of the reasoning. However, using a "meta" prompt that is given the question and the verbose answer and asked to extract just the answer works quite well. This is a cognitive task that would be easier for a person (it's easy to imagine being able to give someone the general task of "read this and pull out whatever the answer is" and have that work across many domains where the user had no expertise, just because natural language is so powerful). So, when writing programs, remember that something that would be hard for a person is likely to be hard for the model, and breaking patterns down into easier steps often gives a more stable result

9 - **Beware "pareidolia of consciousness"; the model can be used against itself."** It is very easy to imagine a "mind" inside an LLM. But there are meaningful differences between human thinking and the model. An important one that can be exploited is that the models currently don't remember interactions from one minute to the next. So, while we would never ask a human to look for bugs or malicious code in something they had just personally written, we can do that for the model. It might make the same kind of mistake in both places, but it's not capable of "lying" to us because it doesn't know where the code came from to begin with. _This means we can "use the model against itself" in some places – it can be used as a safety monitor for code, a component of the testing strategy, a content filter on generated content, etc. _

DEMO #1

PROMPT ENGINEERING

1. Summarization
2. Classification
3. Rewriting
4. Text Generation

CONCLUSIONS

RESOURCES

Azure Open AI Service documentation
https://learn.microsoft.com/en-us/azure/cognitive-services/openai/

Azure Open AI Samples
https://github.com/Azure/azure-openai-samples

Sample: ChatGPT with Enterprise Data
https://github.com/Azure-Samples/azure-search-openai-demo/

Sample: Azure Open AI Embeddings Q&A
https://github.com/Azure-SAmples/azure-open-ai-embeddings-qna

Tool: LangChain
https://docs.langchain.com/docs/

Tool: Semantic Kernel
https://github.com/microsoft/semantic-kernel

THANK YOU