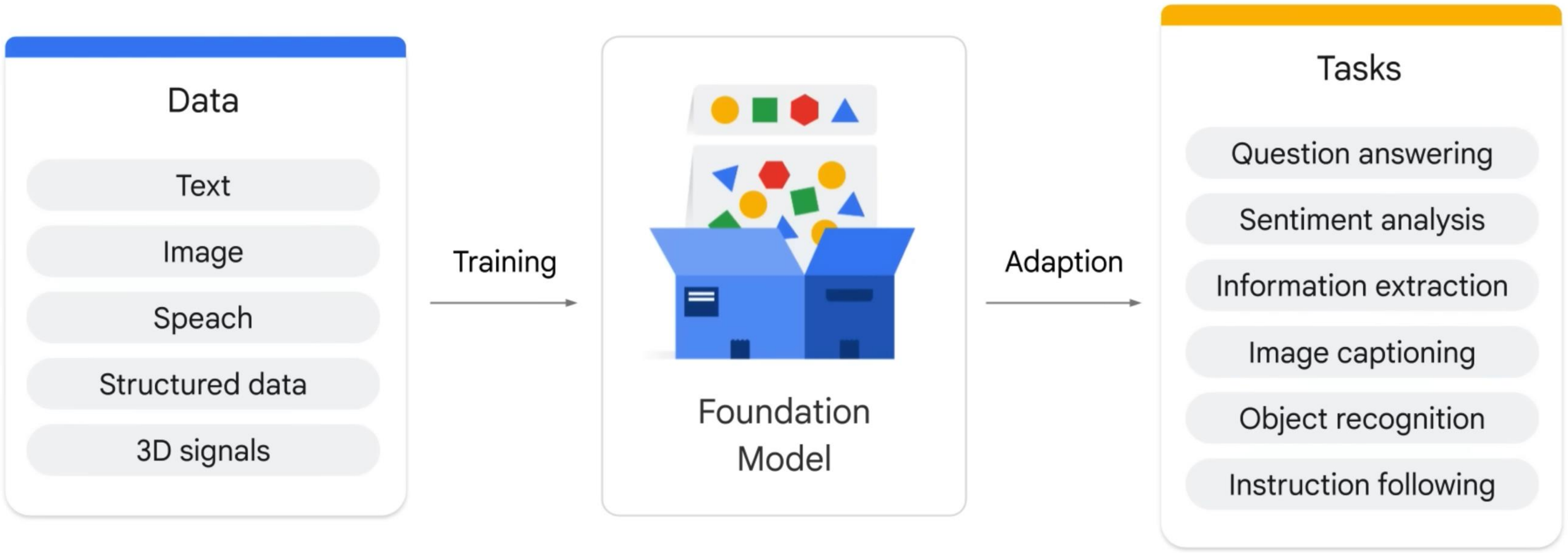
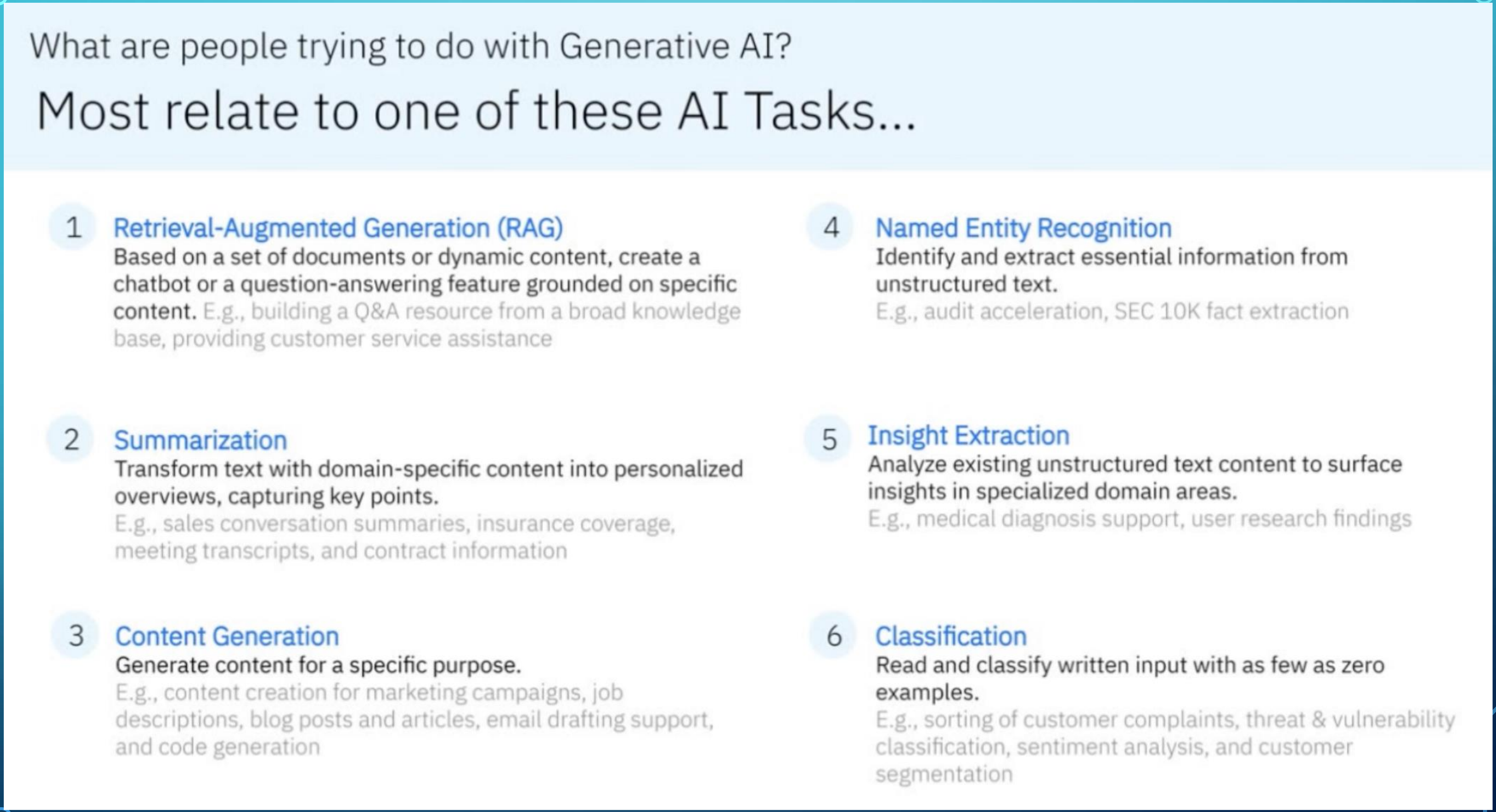
From Tokenization to Embeddings

Week 3


Data
Text
Image
Speach
Structured data
3D signals
Training
Foundation Model
Adaption
Tasks
Question answering
Sentiment analysis
Information extraction
Image captioning
Object recognition
Instruction following



What are people trying to do with Generative AI?

Most relate to one of these AI Tasks...

1 **Retrieval-Augmented Generation (RAG)**
Based on a set of documents or dynamic content, create a chatbot or a question-answering feature grounded on specific content. E.g., building a Q&A resource from a broad knowledge base, providing customer service assistance

2 **Summarization**
Transform text with domain-specific content into personalized overviews, capturing key points.
E.g., sales conversation summaries, insurance coverage, meeting transcripts, and contract information

3 **Content Generation**
Generate content for a specific purpose.
E.g., content creation for marketing campaigns, job descriptions, blog posts and articles, email drafting support, and code generation

4 **Named Entity Recognition**
Identify and extract essential information from unstructured text.
E.g., audit acceleration, SEC 10K fact extraction

5 **Insight Extraction**
Analyze existing unstructured text content to surface insights in specialized domain areas.
E.g., medical diagnosis support, user research findings

6 **Classification**
Read and classify written input with as few as zero examples.
E.g., sorting of customer complaints, threat & vulnerability classification, sentiment analysis, and customer segmentation

TOKENIZATION IS THE FIRST THING THAT HAPPENS

- **Byte Pair Encoding (BPE):** BPE is a subword tokenization algorithm that builds a vocabulary of variable-length subword units based on the frequency of their occurrences in the training data. It's widely used in NLP tasks and has been employed in models like GPT-2.

- **WordPiece:** Similar to BPE, WordPiece is a subword tokenization method. It starts with individual characters as tokens and iteratively merges them based on the likelihood of their co-occurrence in the training data. BERT, for example, uses the WordPiece tokenizer.

- **SentencePiece:** SentencePiece is a more recent subword tokenization method that extends the BPE and WordPiece algorithms. It treats the input text as a sequence of sentences and segments it into a user-specified number of pieces. It has gained popularity in models like T5.

BYTE PAIR ENCODING (BPE)

Let's say we have the word "chatbot" and the following pairs are identified during BPE:

1. ('c', 'h'): chatbot
2. ('cha', 't'): chatbot
3. ('chat', 'b'): chatbot
4. ('chatb', 'o'): chatbot

The final vocabulary might include: {'c', 'h', 'cha', 't', 'chat', 'b', 'o'}.

Used by GPT-2, RoBERTa

WORDPIECE:

Assuming we start with individual characters and iteratively merge based on co-occurrence:

1. 'c', 'h', 'a', 't', 'b', 'o' → 'cha', 't', 'b', 'o'
2. 'cha', 't', 'b', 'o' → 'chat', 'b', 'o'

The final vocabulary might include: {'c', 'h', 'a', 't', 'b', 'o', 'cha', 'chat'}.

Used by BERT, DistilBERT

SENTENCEPIECE

Using SentencePiece, the algorithm might segment sentences into a specified number of pieces:

- Input Sentence: "Chatbots are fascinating."
- Output Pieces: {'Chat', 'bot', 's', ' are', ' fascinating', '.'}

Used by GPT-3

THERE ARE PLUSES AND MINUSES

Word-based Tokenization

- Very large vocabularies
- Large quantity of out-of-vocabulary tokens
- Loss of meaning across very similar words

Character-based tokenization

- Very long sequences
- Less meaningful individual tokens

IT'S A BALANCING ACT

- The goal is to efficiently represent the language's vocabulary with a manageable number of tokens.
- Tokenizers are often tailored to the specific architecture of the model and may be variations of BPE or SentencePiece.
- These methods are widely used in pre-training large language models because they efficiently handle the vast and diverse vocabulary present in natural language. They allow the models to represent and understand the complexity of language by breaking it down into meaningful subword units, collectively known as meaning spaces.

EMBEDDINGS AND NUMERICAL REPRESENTIONS, COMES NEXT

- When a language model is trained, it learns to represent words or tokens as vectors in a high-dimensional space.
- This process is known as word embeddings. The idea is to capture semantic relationships between words by placing them in a continuous vector space, where similar words are closer in the space.

FOR EXAMPLE…

For example, let's consider a simplified two-dimensional space where words are represented as points:

- The word 'Chat' might be represented as the vector (1, 2).
- 'bot' might be represented as (3, 1).
- 'are' might be represented as (0, 5).
- During training, the model adjusts these vectors based on the context in which words appear. So, if 'Chat' and 'bot' often appear together, their vectors might become closer in the embedding space, reflecting their semantic relationship.
- Now, when you tokenize a sentence using SentencePiece or another tokenizer, each token (e.g., 'Chat', 'bot', 'are') is associated with its corresponding embedding vector.
- And finally, you have the following tokens:
- 'Chat' → (1, 2)
- 'bot' → (3, 1)
- 'are' → (0, 5)

GOOD TOKENIZATION IS ESSENTIAL FOR PREDICTING THE NEXT WORD

Embeddings	[203, 45, 1096, 77, 12, 92, 8]
Tokens	[‘John’, ‘s’, ‘pet’, ‘is’ ‘a’ ‘bat’ ‘.’]
Raw text	John’s pet is a bat.

YOU’RE IN MY MEANING SPACE

- The model uses these numerical vectors as input to make predictions. During the generation phase, the model can also convert numerical vectors back into text, allowing it to generate coherent and contextually appropriate sentences.
- This process enables the model to understand and generate text based on the learned relationships between tokens in the embedding space.
- It's how LLMs operate, capturing both syntactic and semantic information about language.

FIRST, THE SENTENCE IS TOKENIZED AND THEN COMES EMBEDDINGS!

Raw text:	First, the sentence is tokenized and then comes embeddings!
Tokens	[‘First’ ‘,’ ‘the’ , ‘sen’, ‘tence’, ‘is’ , ‘ token’, ‘ized’ , ‘and’ , ‘then’ , ‘come’ , ‘s’ , ‘em’ , ‘bed’ , ‘ding’ , ‘s’ , ‘!’]
Embeddings	[203, 45, 1096, 77, 12, 92, 8, 423, 5096, 113, 97, 402, 420, 76, 19, 45]

Embeddings become **vector representations** and similar words gravitate towards each other in the vector **meaning space.**

Context matters, all the other words in the sentence help find meaningful words.

The smaller the **Cosine similarity**, greater the prediction.

YOU'RE IN MY MEANING SPACE

- The model uses these numerical vectors as input to make predictions. During the generation phase, the model can also convert numerical vectors back into text, allowing it to generate coherent and contextually appropriate sentences.
- This process enables the model to understand and generate text based on the learned relationships between tokens in the embedding space.
- They allow the models to represent and understand the complexity of language by breaking it down into meaningful subword units, collectively known as **meaning spaces.**
- It's how LLMs operate, capturing both syntactic and semantic information about language.
- Fewer tokens means fewer computations, but it can also mean, less meaningful spaces!
- It's a balancing act.

YEAH BUT….

- Around 93% of ChatGPT-3 is trained using the English language.
- In Common Crawl, a web scraper from the Internet, English made up 43% of the corpus; European languages accounted for 38%; Chinese and Japanese were 9%, Armenian was not even a rounding error!
- Just so it happens, that all LLMs fare better with "high resource" languages for which training data is plentiful.
- That is a problem for those hoping to export AI to poor countries, in the hope it might improve everything from schools to healthcare to farming.
- So, researchers are trying to make LLMs more fluent in less widely spoken languages, and there are three approaches to this effort:
  - Modify the tokenizers
  - Improve the datasets
  - Tweak models after they have been pre-trained: HFRL (human feedback reinforcement learning) and fine tuning

MODIFY TOKENIZERS, IMPROVE DATASETS, FINE TUNE LLMS

- **Modify tokenizers** in the case of text in Devanagari, a script used with Hindi. When tokenized the standard way, the same text in English needs three to four times more tokens.
- And Indian startup Sarvan AI has written a tokenizer for Hindi, which substantially cuts that number. And their LLM is called **OpenHathi**.

- **Improve datasets** often means digitizing reams of pen-and-paper texts.
- In November 2023, a team of researchers at Mohamed bin Zayed University in Abu Dhabi released the latest version of an Arabic-speaking model called "**Jais**."
- It has 1/6 as many parameters as ChatGPT-3 but performs on par with it in Arabic.

- **Tweaking the models** after they have been pre-trained is another approach.
- Both Jais and OpenHathi have had some question-and-answer pairs hand crafted by humans.
- The same happens with Western chatbots, to stop them spreading what their makers see as disinformation.
- **Ernie Bot**, an LLM from **Baidu**, a Chinese Tech company, has been tweaked to try to stop it saying things to which the government might object.