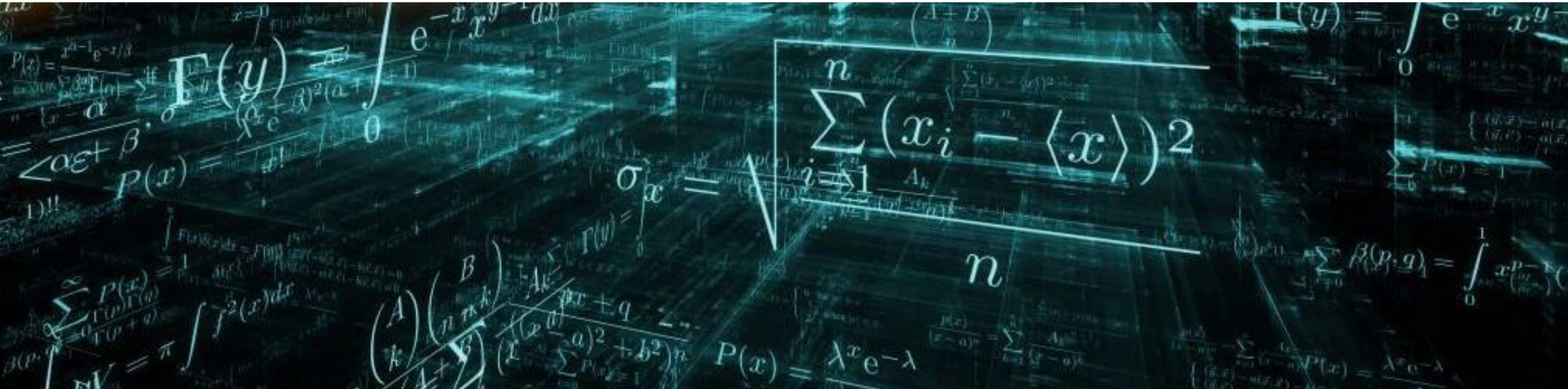
Universität Basel

Computer Science / 15731-01 / 2023

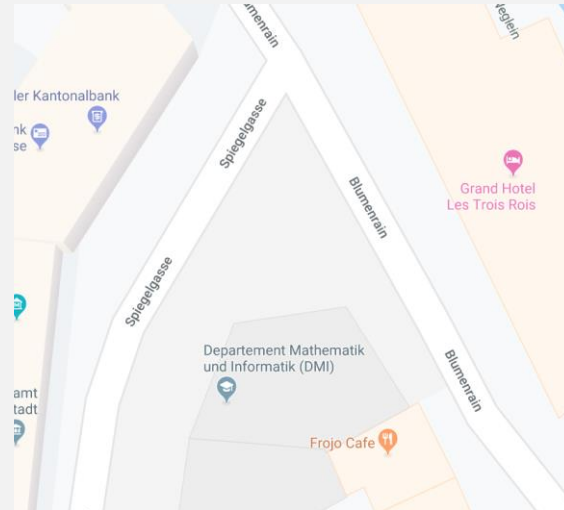
Multimedia Retrieval

Chapter 1: Introduction

Dr. Roger Weber, roger.weber@gmail.com



Course ID	15731-01
Lecturer	Dr. Roger Weber, roger.weber@gmail.com
Time	Friday 15:15 - 18:00 (1st/2nd hour for theory, 3rd hour for exercise & practice → bring your own laptop) Note: changes are announced on web site and / or per e-mail ahead of lectures
Location	Physical presence: Seminarraum 00.003, Spiegelgasse 1 If physical presence is not possible, we use Zoom Meetings. Please check the schedule for updates. During physical presence lectures, no Zoom meetings and no video recordings are available.
Prerequisites	Basics of programming (Python preferred) Mathematical foundations (for some parts)
Content	Introduction to multimedia retrieval with a focus on classical text retrieval, web retrieval, extraction and machine learning of features for images, audio, and video, index structures, search algorithms, and concrete implementations. The course is touching on classical and current information retrieval techniques and search algorithms.
Exam	Oral exam (30 minutes) on January 6, 13, 27 (15.00 – 18.00)
Credit Points	6
Grades	From 1 to 6 with 0.5 steps. 4.0 or higher required to pass exam.
Homepage	WEB: https://dmi.unibas.ch/de/studium/computer-science-informatik/lehrangebot-hs23/15731-lecture-multimedia-retrieval/ ADAM: https://adam.unibas.ch/goto_adam_crs_1547405.html All materials are published in advance. Practical exercises to be submitted to ADAM



Structure of the class

Foundation	1 Introduction	We cover motivation, a summary of history, the generic retrieval process and its variations, a quick overview of metadata, and view demos to get us started
	2 Evaluation	We focus on evaluating and comparing retrieval systems and machine learning approaches. This serves as the basis for assessing the effectiveness of the methods covered in most of the chapters
	*11 ML Methods**	We cover essential machine learning methods as needed for content analysis and the extraction of metadata items. As we progress through the course, we will visit individual chapters as need
Text & Web Retrieval	3 Classic	We explore classical text retrieval models, with a particular emphasis on vector space retrieval. We also delve into Lucene, OpenSearch and Elasticsearch which showcase the capabilities of these models
	4 Advanced	We examine natural language processing using NLTK as an example. Additionally, we explore contemporary methods for creating embeddings and leveraging generative AI to improve results
	5 Web & Social	We focus on web and social media retrieval, particularly examining methods to influence rankings based on the relationships between documents
	6 Vector Search	We explore the challenge of searching through embeddings and feature vectors. We discuss the "curse of dimensionality" and study contemporary techniques used by products like Lucene, OpenSearch, and Elasticsearch
Image Retrieval	7 Basic	We cover the human perception of visual signal information and examine several algorithms for extracting features that describe color, texture, and shape aspects found in the images
	8 Advanced	We delve into neural networks and explore the concept of deep learning. We apply these techniques to extract higher-level features, including classifications, facial recognition, and object bounding boxes
Audio Retrieval	9 Basic	We cover the human perception of audio signals and study various algorithms for extracting features in both the time and frequency domains. Additionally, we delve into the unique case of extracting musical features
Video Retrieval	10 Basic	We discuss fundamental elements of motion detection and video segmentation. Specifically, we focus on identifying shot and scene boundaries in videos

For exams, Chapter 11 requires the understanding of high-level concepts, basic formulas, and algorithms for each method. It's important that you can explain their applications for retrieval and discuss their strengths and weaknesses. You don't need to memorize formulas and algorithms beyond the basics concepts.

Timeline and Organization of the course

Date	Theory: 15:15 / 16:15	Practice: 17:05	Where*
Sep 22	1 Introduction	Python, Jupyter Notebook	University*
Sep 29	2 Evaluation	Ex 1, deep dive topics	University*
Oct 6	3 Classical Text Retrieval	Q&A, deep dive topics	tbd
Oct 13	3 Classical Text Retrieval	Ex 2, Q&A, deep dive topics	tbd
Oct 20	4 Advanced Text Retrieval	Q&A, ML process, neural networks	tbd
Oct 27	4 Advanced Text Retrieval	Ex 3, Q&A, transformers	tbd
Nov 3	5 Web and Social Retrieval	Prep Exam	University*
Nov 10	6 Vector Search	Ex 4, Q&A, deep dive topics	University*
Nov 17	7 Basic Image Retrieval	Ex 4, Q&A, convolutional networks	University*
Nov 24	*No Lessons (Dies Academicus, last Friday in November)*		
Dec 1	8 Advanced Image Retrieval	Ex 5, Q&A, deep learning	University*
Dec 8	9 Basic Audio Retrieval	Ex 5, Q&A, deep dive topics	University*
Dec 15	10 Basic Video Retrieval	Q&A, deep dive topics	University*
Dec 22	(11 ML Methods) – covered in the chapters we need them	Eval & Prep Exam	University*

- **Theory:** please study the material in advance. During the lessons, we will cover the essentials with demos and discussions, but some details will be omitted for self-study and to maintain a good pace. Refer to the schedule and announcements in ADAM. As a general rule, aim to read about 30-40 pages ahead
- **Practice:** during the 3rd hour, we engage in both theoretical and practical exercises. We explore Python and software packages relevant to the retrieval topics we cover. This part is optional, but active participation can benefit you in the exams (more details on the next page). No special materials are provided, and whenever possible, public tutorials will be used to introduce software packages. Feel free to join with a curious mindset and ask questions.

* University: Spiegelgasse 1, Seminarraum 00.003, **no zoom available**, **no video uploads** after lecture
* Zoom: see meeting link on Web / in ADAM, video uploads after lecture

Exams and how to prepare for them

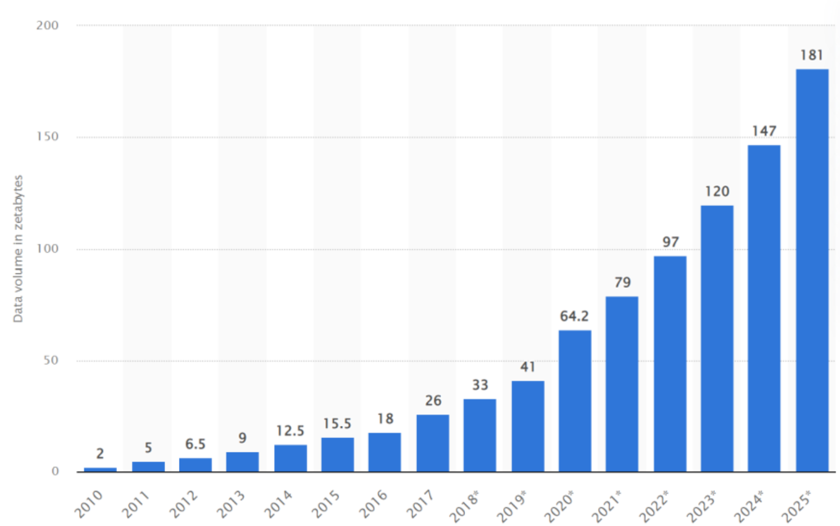
- Exams are scheduled in the first weeks of January
  - each student will have a 30-minute slot
  - slots will be assigned in December based on your preferences and available slots
  - we will cover three topics, with each topic allocated around 8-10 minutes
  - each topic will include several questions of increasing difficulty, so be accurate and fast to earn maximum points
- Prerequisites for Exams: all exercises are optional, but practical exercises can help to round-up exam grades
  - you don't need to submit theoretical exercises; we'll provide and discuss solutions during the 3rd hour
  - you can submit practical exercises but you won't receive grades; **instead you earn points for the exam**
  - **you do not earn points for theoretical exercises**
  - points will be converted into an **upgrade of your grade ranging from 0 to 0.3**
  - the upgrade is added to your oral exam grade for the final result (rounded to the nearest 0.5 grade step)
  - Examples:

1. Student submits <u>some practical exercises</u>, earning an upgrade of 0.2. In the oral exam, the student doesn't perform well and receives a grade of 3.6. Final result: 3.6 + 0.2 rounded up to 4.0 (pass)
2. Student earns an upgrade of 0.3 for <u>very good submissions</u>. In the oral exam, the student receives a grade of 5.5, due to some difficulty with challenging questions. Final result: 5.5 + 0.3 rounded up to 6.0 (pass, excellent)
3. Student <u>doesn't have time</u>, earns no upgrade, but performs well in the oral exam, receiving a grade of 5.7 struggling only with the toughest questions. Final result: 5.7 + 0.0 rounded down to 5.5 (pass, very good)
4. Student <u>doesn't have time</u>, earns no upgrade, and doesn't perform well in the oral exam, receiving a grade of 3.7 with struggles in many questions. Final result: 3.7 + 0.0 rounded down to 3.5 (failed)

- Submission deadlines:
  - Theoretical exercises: no submissions to ADAM; self correction against distributed solution
  - Practical exercises: submission to ADAM by 31st Dec (23:59); groups of 2 are possible, but you need to provide more / better results to earn points

1.1 Motivation

- Data volumes have experienced significant growth over the past 10 years as illustrated with the figure on the right from Statista.com (annual data volumes created worldwide)

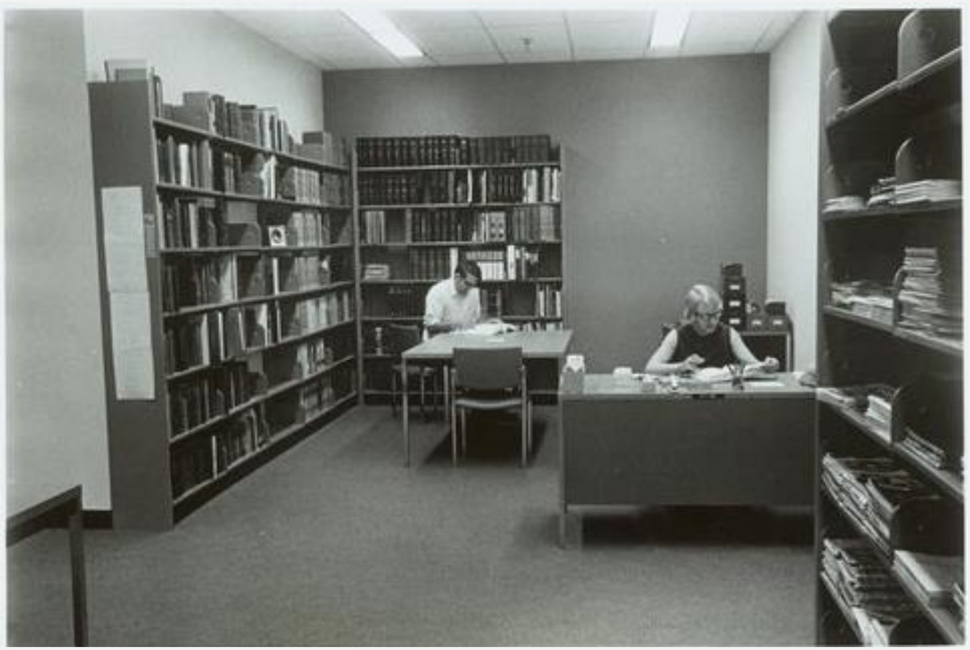


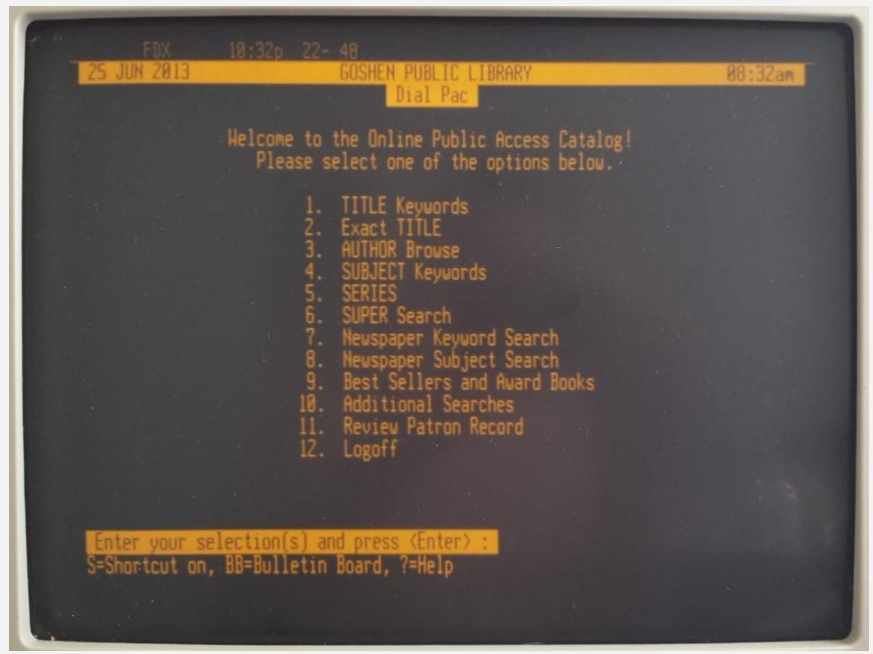
- To put these figures into perspective, 1 zettabyte equals:
  - 1 billion terabytes
  - or the equivalent of 12 billion 4k videos
  - or 1000x all titles on IMDb.com in 4k
- The driving factors behind this growth are social media, IoT devices, data science, research (e.g., pharmacy), streaming and gaming services. For instance, Facebook records approximately 8 billion video views, while YouTube handles 50 billion short views on a daily basis. At YouTube alone, uploads of video material went up from 6 hours per minute in 2007 to 500 hours per minute in 2019. That means every hour, YouTube hosts 30,000 hours of new video material.
- This exponential growth makes it challenging to effectively search through such vast amounts of data:
  - The **sheer volume** of data makes it impractical and inefficient to analyze data manually. Additionally, the number of potentially relevant documents impacts ranking of results. For instance, a simple Google search for ‘ford’ returns 2 billion hits. However, determining which results are relevant to a user's expectations is not easy.
  - Moreover, information is available in **various formats** such as structured, semi-structured, and unstructured data. Navigating through these diverse data types and extracting meaningful insights requires sophisticated techniques and tools. Each additional data domain requires new domain knowledge and technical expertise.
  - Finally, data is generated and updated at an **unprecedented pace**. Real-time and near-real-time data sources such as sensor networks and social media, require efficient retrieval mechanisms that can keep up with the pace. Consider, for example, the short timespan during which a new tweet is relevant. A system that retrieves information with delay soon becomes obsolete as users turn to systems that produce answers in near real-time.
- Over the decades, information retrieval has undergone significant changes driven by ever more challenging requirements and improved search approaches. The following pages provide a short overview.

Decades: 1960s – 1980s

- Users: Researchers, Librarians, Information Professionals
- Use Cases: Literature search (academia), book search (library), bibliographic search (references)
- Key Technologies: Boolean Retrieval, inverted indexes, TF-IDF weighting, manual tagging/keywords
- Retrieval model: Retriever-only, Retriever-Filter, (Retriever-Ranker)
- Limitations: small data sets, slow response times, difficulties to formulate the right query (had to vary keywords and combinations of keywords)

Examples: IBM Stairs, LexisNexis, Integrated Library Systems





Decades: 1990s – 2000s

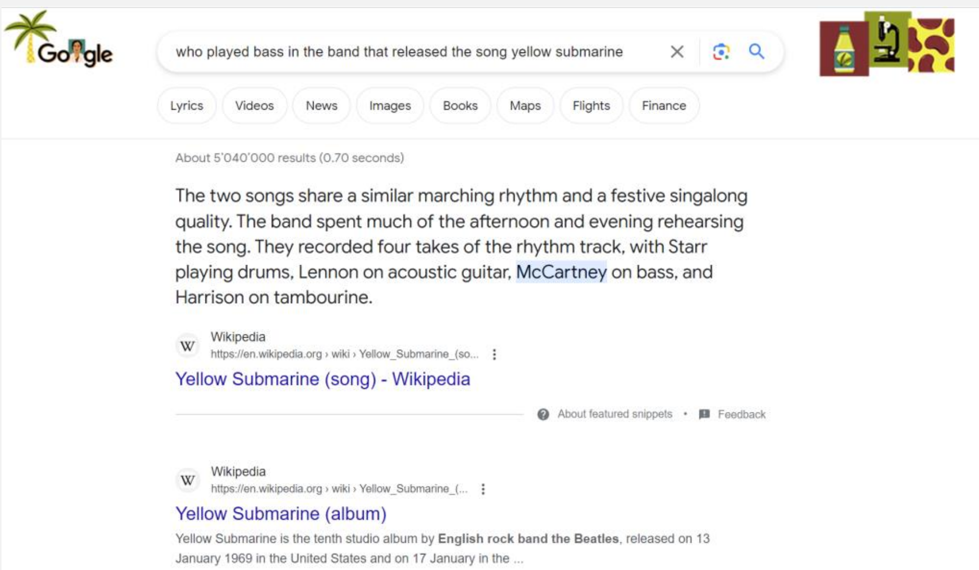
- Users: General public, e-commerce consumers, professionals and business users
- Use Cases: web search, e-commerce search, business/market intelligence, academia
- Key Technologies: Vector Space Retrieval, Probabilistic Retrieval, Web Search Engines, Query Expansion, separation of retrieval & sort
- Retrieval model: Retriever-Ranker, (Retriever-Filter)
- Limitations: data & index explosion due to exponential growth, large retrieval and indexing costs, good recall values but poor perceived precision (i.e., document not relevant for the user), quality of data

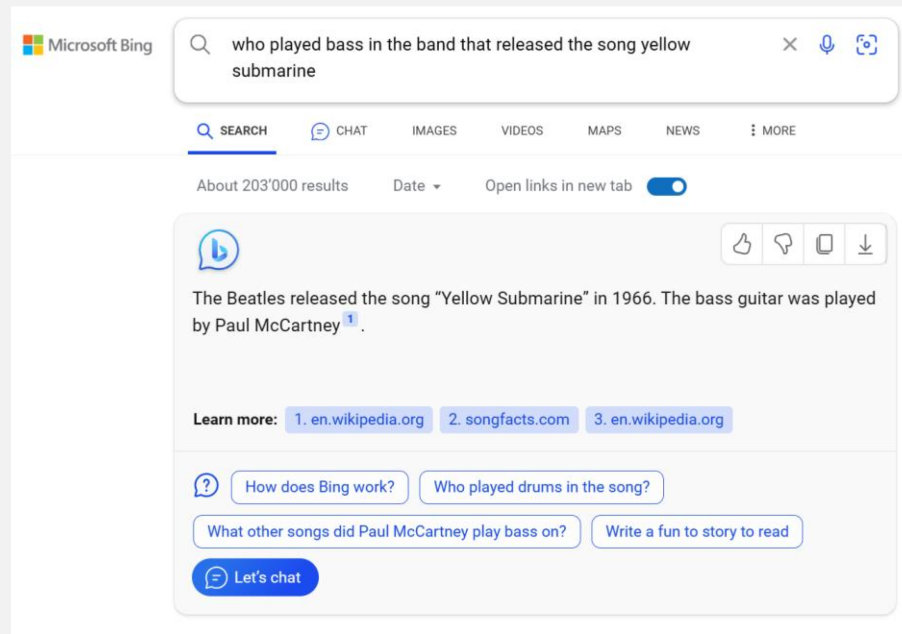
Examples: AltaVista, Yahoo!, Google, Amazon.com, Apache Lucene, PubMed, ProQuest

Decades: 2010s – present

- Users: Mobile users, social media users, e-commerce consumers, business users, data scientists
- Use Cases: personalized web, social & e-commerce search, consumer & market analytics, academia
- Key Technologies: semantic search (NLP), embeddings, personalization & contextualization, retrieval augmented generators (RAG), chat bots
- Retrieval model: Retriever-Reader, Retriever-Generator
- Limitations: data & index explosion due to exponential growth, large retrieval and indexing costs, quality of data, transparency of filtering / sorting / generated answers, (near) real-time updates

Examples: Google, Bing, Amazon.com, Spotify, Facebook, Twitter, TikTok, Apple Siri





- Searching for images, audio files, and videos is much harder due to the so-called **Semantic Gap**
  - Users typically ask systems with keywords (either typed or spoken, e.g., Alexa). This works well with text documents as the system can match query terms with terms found in documents (Alexa transcribes spoken text to text before querying)
  - For images, as an example, the system is not able to match keywords with pixel information. In the example below, the system's representation of the cat (right hand side) differs from the keyword representation of the user (left hand side)
- If we query in one representation, how can we translate the tokens into the other representation?

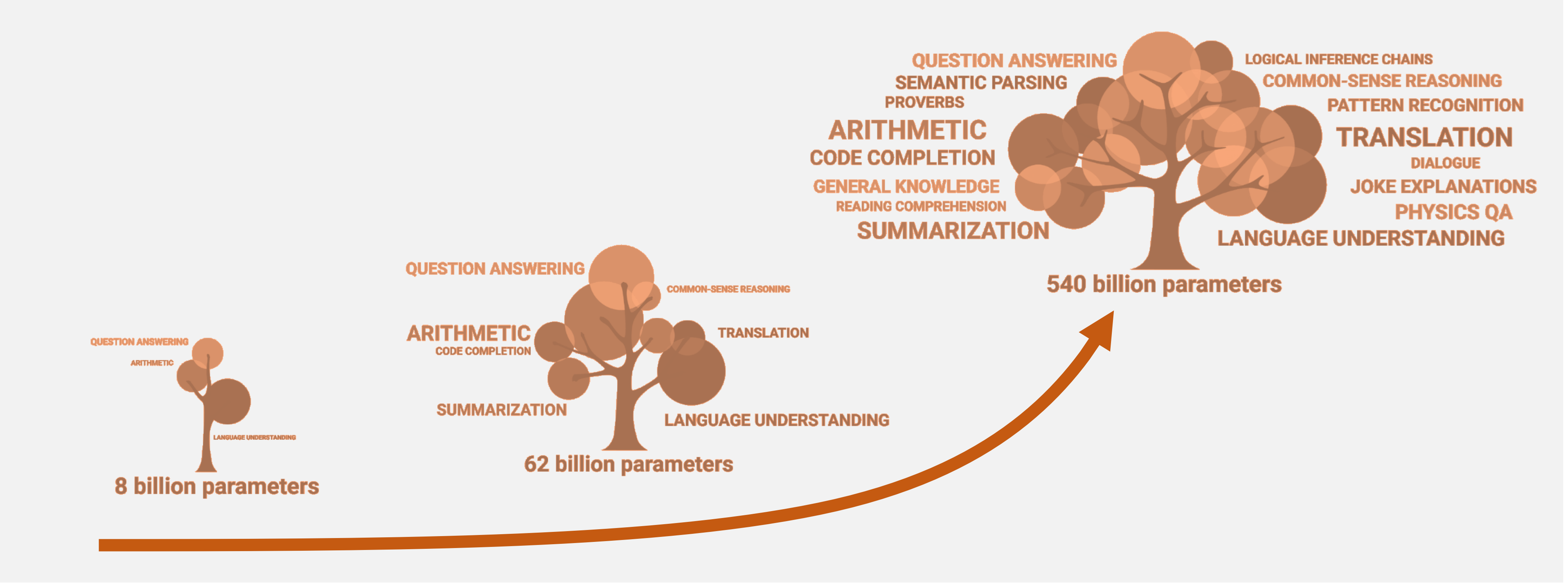
- **Definition: Semantic Gap**

  > The semantic gap refers to the disparity between low-level features extracted from multimedia data and the high-level semantics that humans associate with that data

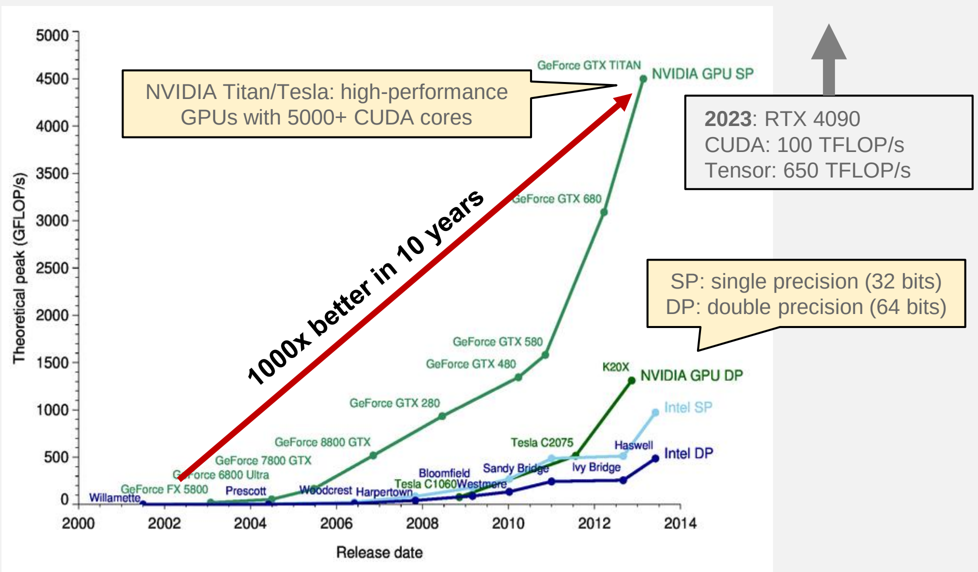
- **How to overcome?**
  - 1960s – 1980s: manual tagging and keyword annotations; inversion of query (store everything about a prominent person in a dedicated folder)
  - 1990s – 2000s: Emergence of large annotated media/metadata archives often with collaborative efforts (IMDb, AllMusic, MusicBrainz)
  - 2010s – present: Emergence of AI technology to automate key word generation and context analysis to improve relevance ranking

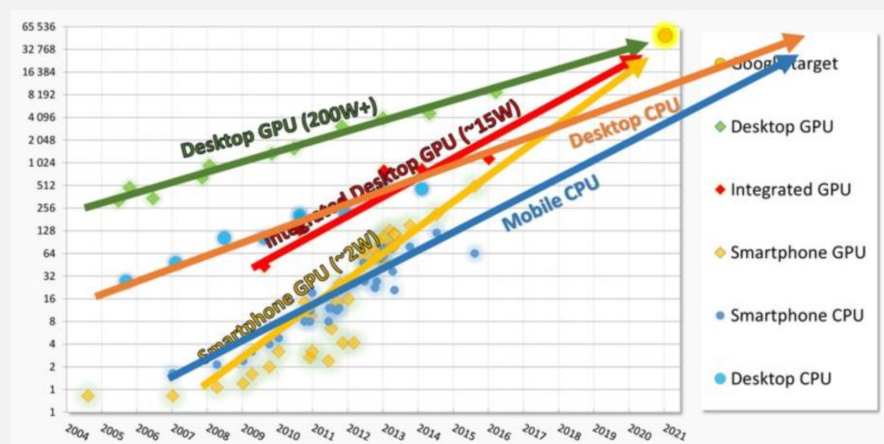
- Improved memory and compute have played a crucial role in breaking through with some of the retrieval challenges in the past decade. For instance, the largest languages models feature up to 550 billion parameters (that is 4.4 TB of uncompressed parameter data for the trained model). This imposes several hard-to-solve challenges:
  - High inference costs to generate next token (550 billion operations, TB of model data in memory)
  - High training costs to process a large corpora of text documents (parameter optimization)
  - High optimization efforts to fine-tune model parameters (hyperparameter optimization)
  - High fine-tuning costs to adopt to specific tasks or to incorporate new text documents
- On the other side, the recent success of GenAI was only feasible with the increase of model parameters allowing AI models to perform tasks at (and sometimes above) the level of humans. Google Research has illustrated the capabilities of AI models in relation to the number of parameters:

source: https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html



- The biggest improvement over the past ten years was the creation of CUDA, an extreme parallel computing platform created by Nvidia. In combination with new neural network algorithms and the advent of map/reduce as a generic distributed computing paradigm, enormous amounts of data became processable through the sheer brute force of 1,000s of connected machines.
- The AMD EPYC 9654 CPU has 96 cores and 192 threads, delivering nearly 1 TIntOps/sec with integers and 550 GFlops/sec with floating point math at 400W power consumption. Assuming a 550B parameter model with 2 operations per parameter for each token (forward and backward step), training 1 million tokens would take approximately 12 days per epoch. Several epochs (let's say 10) are typically needed to optimize the model, requiring around 120 days. Similarly, several runs (let's say 10) are conducted to optimize the hyperparameters, taking approximately 1200 days. To reduce the learning time to a week, we need about 200 such CPUs and a high-bandwidth connection between them.



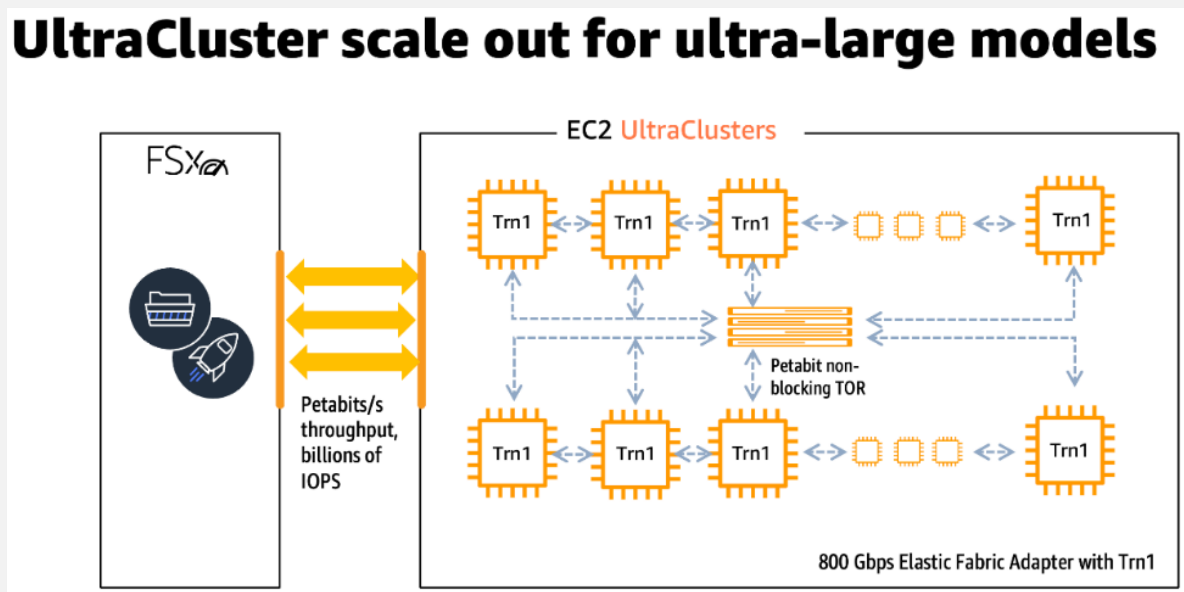


- GPUs like the Nvidia RTX 4090 (5nm) offer up to 100 TFlops with CUDA architecture and 450W power consumption, enabling learning speeds 200 times faster than CPUs at similar power usage. One GPU is able to learn the 1 million token data set within a week's time. On the other side, GPUs allow us to scale the training data set: each additional GPU gives us the ability to train 1 million more tokens. With a 1000 GPUs, we could train 1 billion tokens within a week.
- The Nvidia RTX 4090 features tensor processing units (TPUs) with even higher compute capacities, reaching up to 650 TFlops. Purpose-built TPUs by Google offer 275 TFlops, and Amazon EC2 Trainium and Inferentia accelerators provide about 210 TFlops optimized for training and inference tasks. These options give us 3-6 times faster learning (and inference) than GPUs with similar power consumption. With the Amazon EC2 Trn1.32xlarge instance (16 Trainium accelerators), we get 3.4 petaflops at $10/hour costs and could train 1 million tokens in under 5h (=$50). With a cluster of 360 Trainium accelerators, training 1 billion tokens within a week becomes possible (=$50,000)
- However, the largest language models are trained with more than 1 trillion tokens. Even the largest cluster, the Amazon EC2 UltraCluster with 30,000 Trainium accelerators and 6 ExaFlops of compute performance, will take weeks to months to train on such large data sets

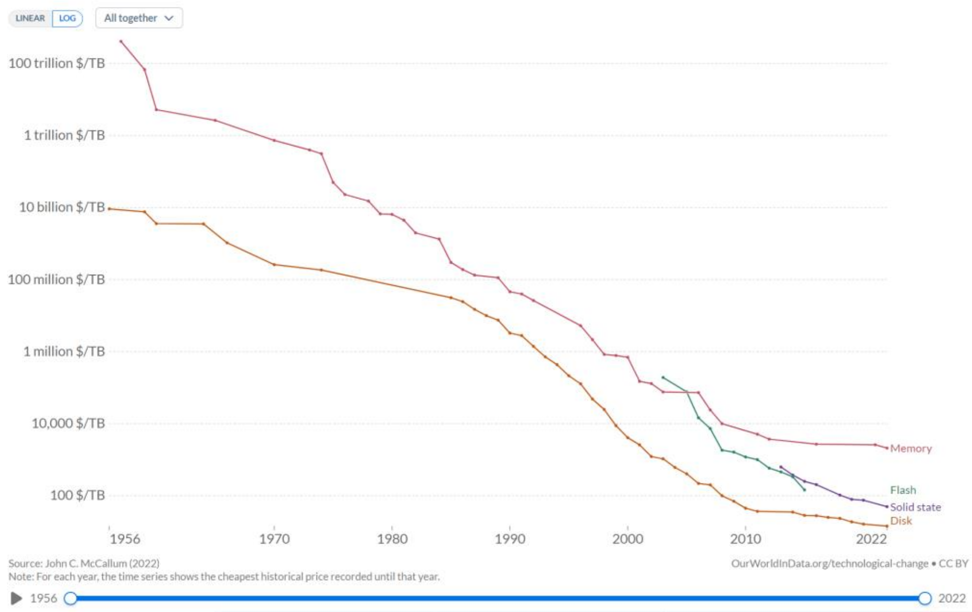
Note: algorithms and regularizations like drop outs significantly reduce compute requirements for large models.

Note 2: the above calculations are rough estimates and technology advances very quickly in this area

Consider [Wikipedia LLM] for a table of PF-days to train recent models. The largest models require 85,000 PF-days, which turns into $6m compute costs with the 3.4 PF of EC2 Trn1.32xlarge and 25,000 days to train. The UltraCluster (costs unknown) still requires 2 weeks to train.



- Illustration of price and space compression of storage



source: https://ourworldindata.org/

In the past decades, we have seen price drops of 50% every 14 months. Every 4 years, the costs decreased by an order of magnitude. On the other side, firms still spend the same amount of $ to increase and replace their storage real estate. As a consequence, the amount of managed storage also grew exponentially and makes it ever more difficult to find relevant information.

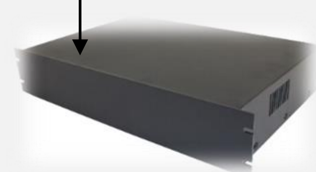
Year	1 TB disk	1 TB memory
1960	$3,600,000,000	$5,240,000,000,000
1970	$259,700,000	$734,000,000,000
1980	$95,000,000	$6,480,000,000
1990	$3,270,000	$46,000,000
2000	$4,070	$700,000
2010	$45	$5,100
2020	$16	$2,600

42U Rack

1 Petabyte of storage (2015)

21x smaller in 3 years

72x 16TB



2U Appliance

1 Petabyte of storage (2018)

- So, how long does it take to read 1 Petabyte? All data points as of 2017:
  - The fastest hard disk have about 200MB/s read rate (and almost same write rate)
  - The fastest solid state disk have about 550MB/s read rate (and 10% smaller write rate)
  - The fastest M.2 flash drives have about 3500MB/s read rate (and 2100MB/s write rate)
  - USB 3.0 can handle up to 640MB/s transfer rate
  - PCI-E 2.0 can handle up to 4GB/s transfer rate
  - 100GB Ethernet can handle up to 10GB/s transfer rate
  - Fibre full duplex 128GFC can handle up to 13GB/s (per direction)
  - The largest Internet exchange point (DE-CIX) operates at up to 700GB/s (average is 430GB/s)

Time to read

Device / Channel	GB/s	1 GB	1 TB	1 PB	1 EB	1 ZB
Hard Disk	0.2	5s	83m	58d	158y	158’000y
Solid State Disk	0.55	1.8s	30m	21d	58y	57’000y
M.2 Flash Drive	3.5	0.3s	5m	80h	9y	9’000y
USB 3.0	0.64	1.6s	26m	18d	50y	50’000y
PCI-E 2.0	4	0.3s	4m	70h	8y	7’900y
Ethernet 100GB	10	0.1s	100s	28h	3y	3’200y
Fibre 128GFC	13	0.08s	77s	22h	2.5y	2’400y
DE-CIX	700	1.4ms	1.4s	24m	16d	45y

- We can’t beat physics…but we can apply brute force with extreme scale and parallelism. A million machines can help to shorten times if the algorithm does scale. Today, large compute clouds have between 1-5 million servers.

1.2 Generic Retrieval Process

- A retrieval system addresses the following problem:

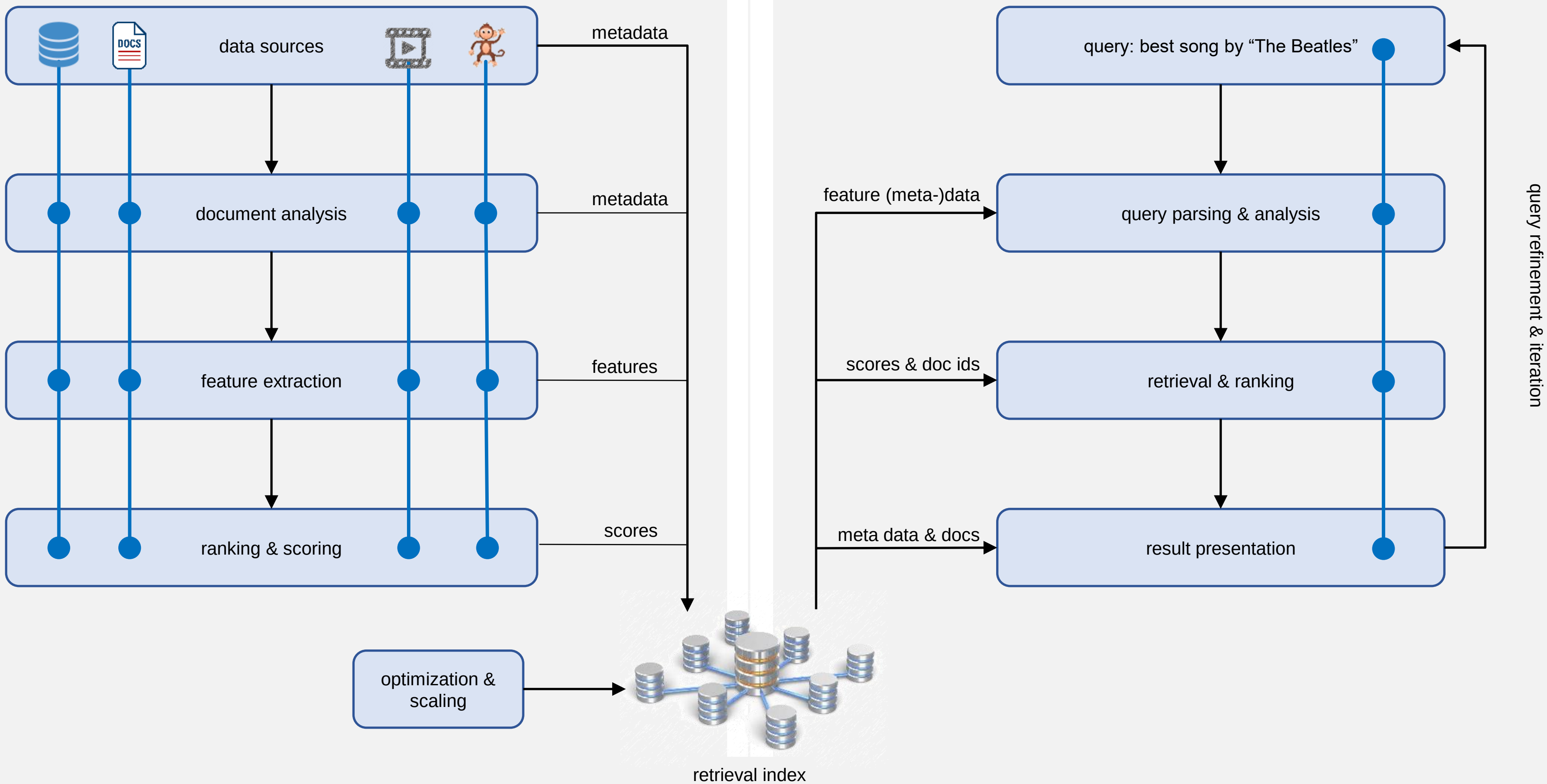
> Given a set of N documents D_0 to D_{N-1} and a query Q, find a set of documents D_{i_j} with $0 \leq j < k$ that are relevant for the query Q in the context of query originator. Rank the documents such that D_{i_0} is the most relevant document and $D_{i_{k-1}}$ is the least relevant document for query Q in the context of the query originator.

- First, what do we mean with “relevant for query Q in the context of the query originator”
  - Relevancy refers to the degree of correspondence or significance between information retrieved and the query. It may further encompasses how well the retrieved information aligns with the user’s needs or intention. Examples:
    - “where can I get a pizza tonight?” requires the location of the user to provide relevant results. The Pizzarella Fantastica may serve the best pizzas, but if you are not close-by, then this result is not relevant
    - many social media services provide information without an explicit query (you may give one if you like). Still users expect relevant information from their social network and not information from a random user
    - you are looking for a product to buy: relevancy depends whether you first want to evaluate your purchase (is the product matching my needs) or whether you want to find the best place to buy it
  - Objective relevance focuses on factual alignment. For instance, if someone searches for "capital of France“, Paris would be objectively relevant due to its status as the capital city. Subjective relevance, however, considers personal preferences. When searching for “movies to watch" subjective relevance would vary depending on individual tastes and interpretations of what constitutes a good movie
    - Search engines utilize feedback loops to improve relevancy. For example, when users click on a search result and spend more time on that webpage, it indicates that the result was relevant. By analyzing such feedback, search algorithms learn to prioritize more relevant results, enhancing the overall search experience.
    - Query-less search refers to a search approach where users can discover content without explicitly entering a search query. Platforms like Instagram Reels and TikTok use algorithms to analyze user preferences, behavior, and trends to recommend relevant videos, enabling users to explore content tailored to their interests without the need for explicit queries.

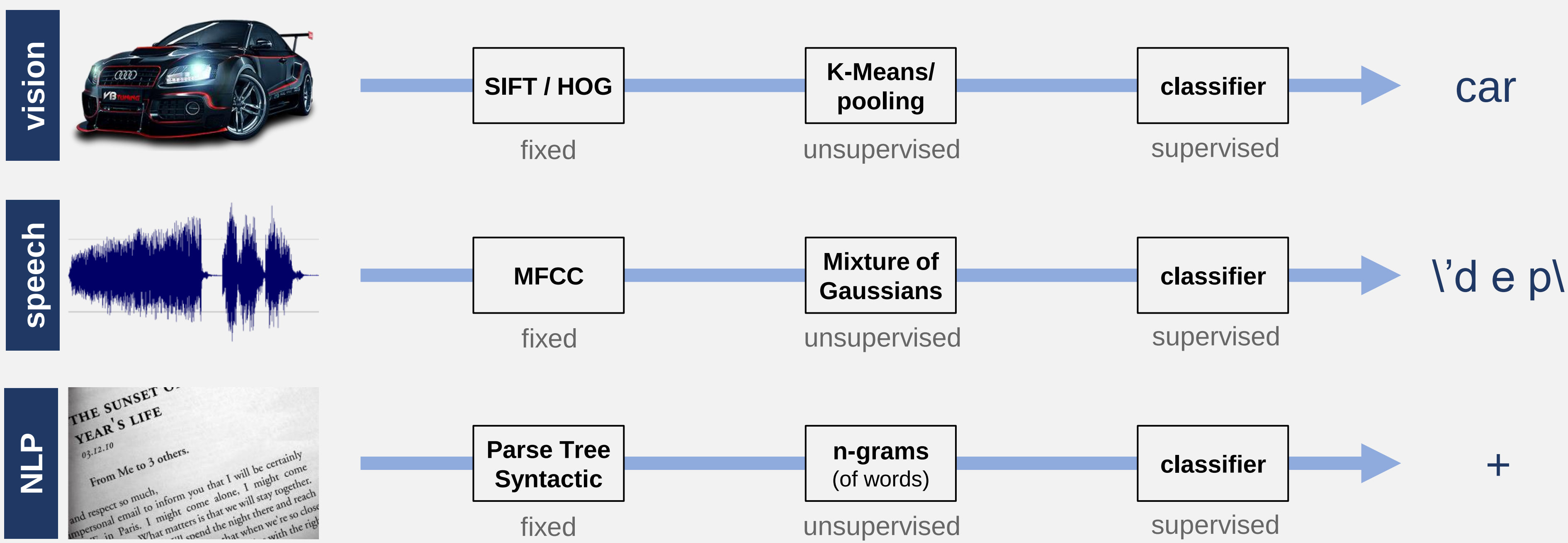
- Second, given the vast amount of data, we cannot scan through the documents at query time as users expect fast responses (well below 1 second). Instead, we split the retrieval process into two parts:

Offline processing: analyze documents before hand, extract meaningful information (so-called features), organize these features in a way that allows for fast retrieval (indexing). Some systems also adjust their (objective) relevance ranking during this phase (see IDF or Google's PageRank later in the course)

Online query answering: parse the query and extract features like for the documents (e.g., embeddings), retrieve relevant documents, score and rank them, and present to the query originator. Optionally: refine the query and iterate

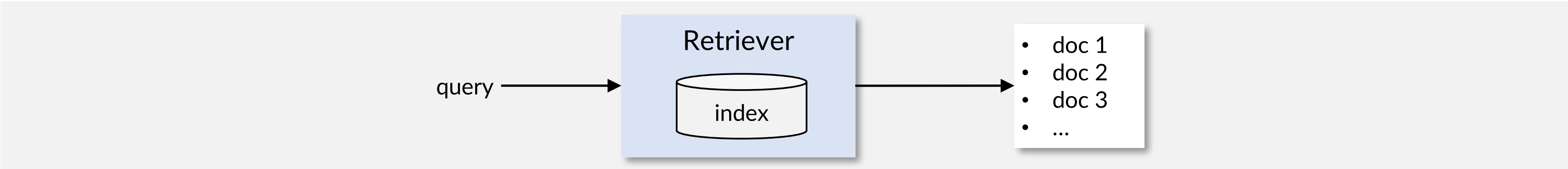


- **Offline Processing:** in the next chapters, we will extensively look at different ways of how to analyze documents
  - text retrieval: extracting term vectors (set of words, bag of words, n-grams), natural language processing (NLP, stemming synonyms, homonyms), extracting embeddings (classic and modern ways to reduce dimensionality of term vectors), discriminative power of terms (IDF), classification
  - web/social retrieval: additional weighting and extraction of key words, link analytics (PageRank), topic search
  - image retrieval: simple features (color, texture, shape), neural network based features (classification)
  - audio retrieval: simple features (frequency domain, amplitude domain), musical features (pitch, tempo, beats), classification, speech recognition (neural networks)
  - video retrieval: shot detection, movement detection
  - …and in this chapter, we start with metadata annotations and extractions



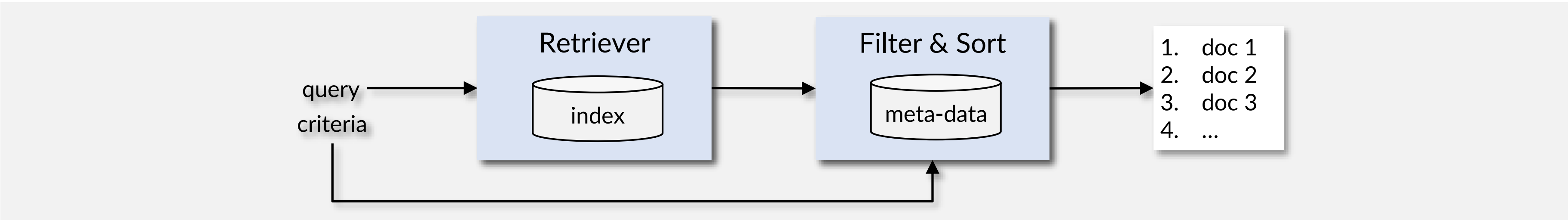
- **Online query answering**: throughout this course, we will study various models from the classical text retrieval models to the most modern approaches that use generative AI. At this point, we discuss the different model types and what differentiates them. Most of these models are still in use and you will recognize them easily as we go through the course:
  - **Retriever-only:** early retrieval systems were built upon this fundamental model. The "retriever" component identifies the relevant documents based on a query and presents them to the user for further examination. While the results can be sorted based on various criteria (id, name, meta data), there is no explicit relevance based ranking. This basic search functionality is commonly available in nearly every operating system (file search) or application that deals with data items, including web-based applications

    → **Example:** https://www.goodreads.com/search?q=agatha+christie&search_type=books



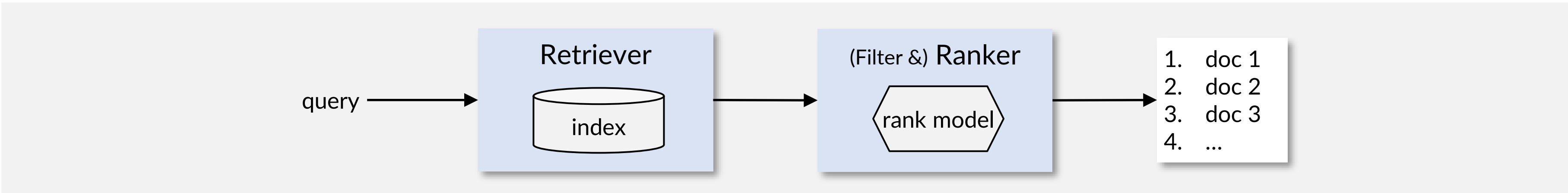
  - **Retriever-Filter:** this model shares similarities with the "Retriever-only" approach but incorporates an additional filter and sorting component to refine the results before presenting them to the user. Filters allow users to narrow down the search results based on input parameters. Additionally, users can influence the sorting of the list by selecting pre-defined attributes or meta-data items, such as sorting by year or sorting by rating. While relevance to the query may impact the result order, other criteria such as popularity, price, or ratings typically dominate the displayed order. This model provides users with enhanced control and customization over the search experience.

    → **Example:** https://www.galaxus.ch/en/search?q=clothes+iron



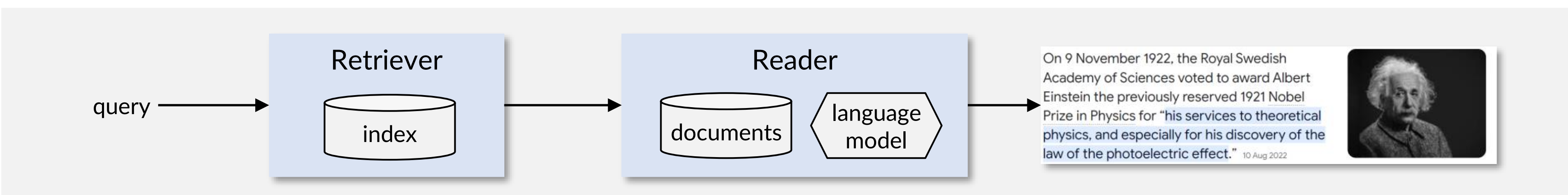
– **Retriever-Ranker:** the “retriever” component selects a pool of candidate documents from the index that match to the query, and the “ranker” component assigns a relevance score to each candidate and returns documents in order of this score. This is a very common model in classical (and modern) retrieval systems, often improved with advanced (semantic) search capabilities and context-sensitive ranking (location of the user, objective importance, subjective importance). Most web search engine use this model to augment the underlying (generic) text retrieval component with web specific ranking. The difference to the previous models is that the user typically can not influence the order of document presentation (except for some pre-defined filtering)

→ **Example:** https://www.google.com/search?q=multimedia+retrieval+lecture (change your location with a VPN client and submit again)



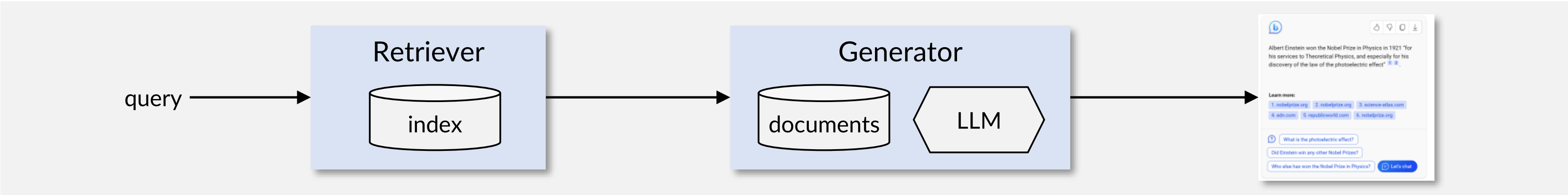
– **Retriever-Reader:** the first modern retrieval model in this list that emerged with the latest generative AI capabilities. This model is used for queries in the form of a question (“what did Albert Einstein win the Nobel Prize for?”). The “retriever” component fetches documents relevant to the query. The “reader” then identifies one or more passages within these documents that answer the question and returns this to the user (instead of listing documents). The reader often uses a language model to identify the passage answering the question.

→ **Example:** Google’s ‘featured snippet from the web’ at the top of search results for question-like queries
https://www.google.com/search?q=What+did+Albert+Einstein+win+the+Nobel+Prize+for%3F
https://www.google.com/search?q=sherlok+holmes+a+study+in+scarlet%3A+who+is+killed%3F



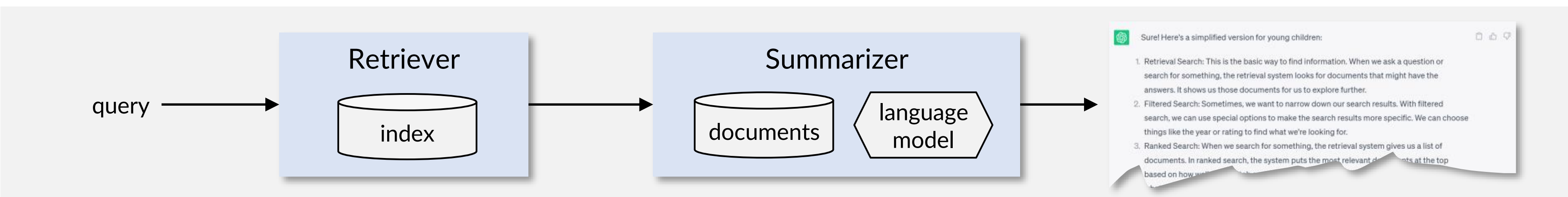
- **Retriever-Generator:** this emerging model harnesses the capabilities of generative AI, also referred to as “Retrieval Augmented Generation” [RAG]). Similarly to the previous model, the retriever component selects relevant documents (and passages) on a query in question format. However, instead of extracting a passage, the snippets together and the query are combined into a “prompt template” for a large language model (LLM). The LLM then generates a comprehensive answer. To put it differently, imagine taking the snippets obtained from a web search and inputing them into ChatGPT along with your question, allowing it to generate a response.

  → Example: Bing chat (“What did Albert Einstein win the Nobel Prize for?”)
  https://www.bing.com/search?q=What+did+Albert+Einstein+win+the+Nobel+Prize+for%3F (only works in Edge with ‘new bing’)



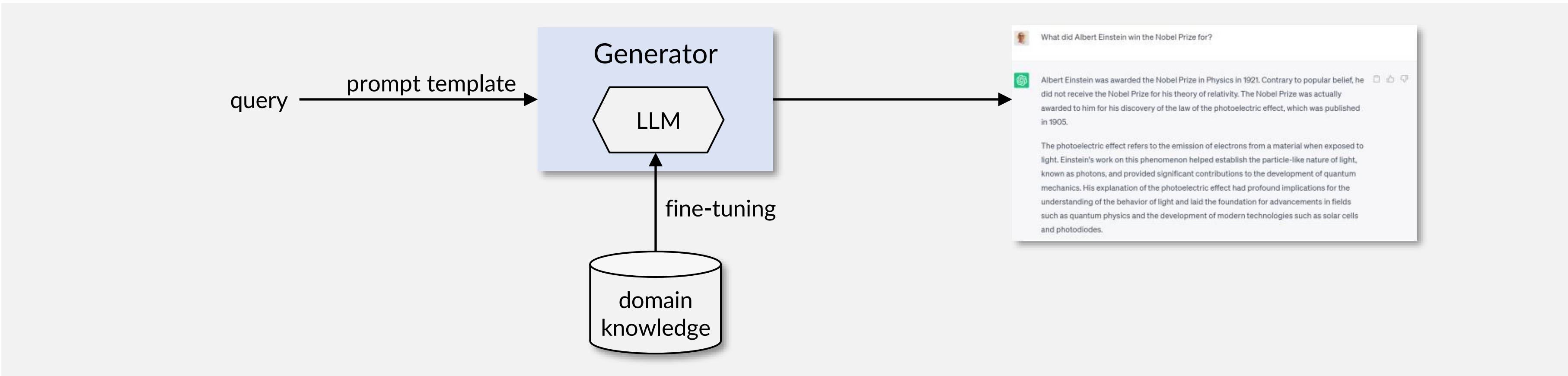
- **Retriever-Summarizer:** in this model, the setup is similar to the ‘Retriever-Generator’ approach. However, instead of generating a specific answer, the large language model (LLM) is instructed to produce a comprehensive summary of the relevant documents. For example, let’s say you want to understand the concept of ‘vector space retrieval’. The retriever first fetches a set of, let’s say, 10 relevant documents. Then, the LLM model of the ‘summarizer’ component generates a condensed summary of these documents. This summary provides a more concise overview of the key concepts than a simple question-answer interaction and allows for further prompt engineering to match the expectation of the user (“explain for young children”)

  → **Example:** no public engine found; let me know if you have a link; try ChatGPT to summarize the contents of a few text snippets



- **Generator-only:** this approach solely relies on generative AI to produce answers to queries. Unlike the previous models, it does not include a retriever component. As a result, the generated answers are limited to the data on which the large language model (LLM) was trained. Prompt engineering enable the model to perform various pre-trained tasks such as text generation, text summarization, and translation. While generic models like GPT-3/4 can provide reasonably good answers to a broad range of questions, fine-tuned models are required for queries that demand specialized domain expertise or that are specific to a business context (e.g., insurance, health, legal)

  → Example: ChatGPT (“What did Albert Einstein win the Nobel Prize for?”)

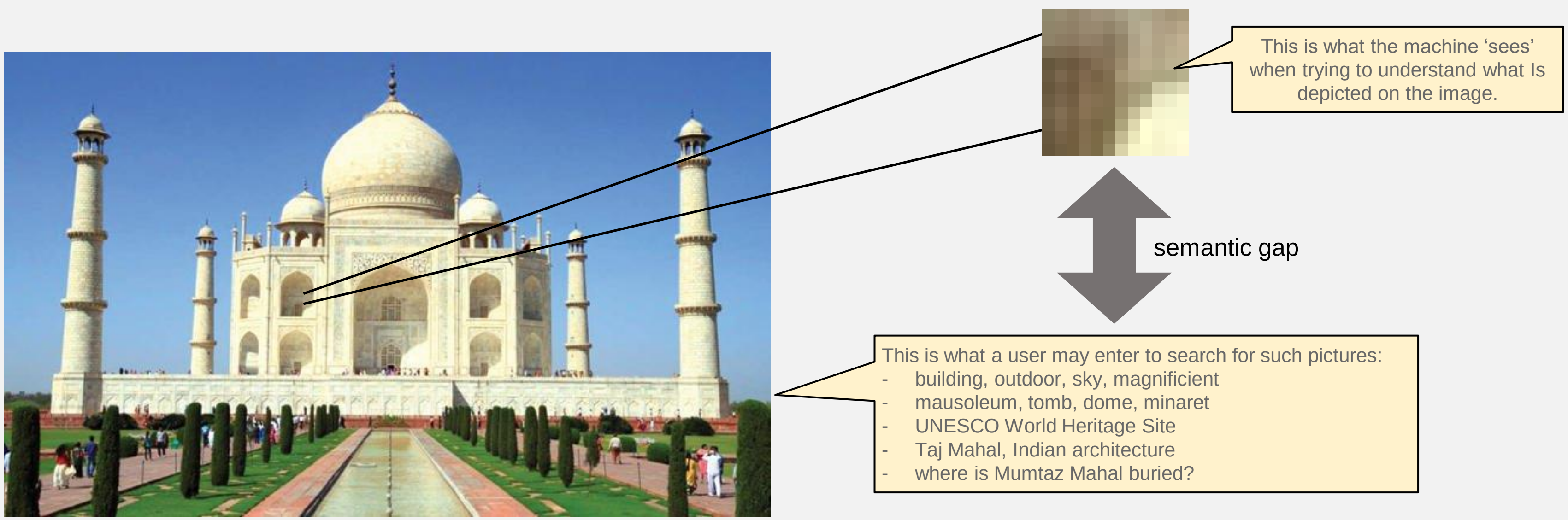


1.3 Metadata and How It Can Help

- We have previously discussed the concept of the "semantic gap" and its challenges. Let's look at an example in the area of image retrieval and consider the picture below of the Tay Mahal as a running example:
  - The context of the picture is as follows:

    The Taj Mahal, located in Agra, India, is a magnificent mausoleum built by Emperor Shah Jahan in memory of his beloved wife Mumtaz Mahal, who passed away in 1631. The Taj Mahal is one of the most famous landmarks in the world and is recognized as a UNESCO World Heritage Site. Mumtaz Mahal's tomb is situated in the main chamber, alongside Shah Jahan's tomb.

  - When users search for pictures of the Taj Mahal, they may use various keywords, as depicted in the lower right box. The image retrieval system must then find matches to these search queries within its image database. However, it faces a challenge as it cannot directly compare the pixel information of images with the keywords provided by the users. Unlike text retrieval, this disparity requires other methods to bridge the semantic gap.
  - To address this gap, we need to map the distinct perspectives (pixels in images and user-provided keywords) into a comparable space where we can more effectively assess the relevance. Throughout this course, we will elaborate on various approaches to achieve this, and focus in this introductory chapter on meta data in general.

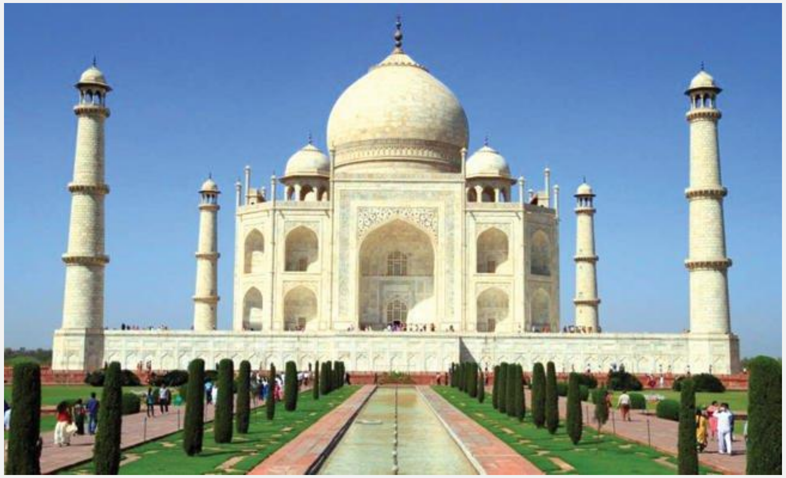


- Meta data refers to additional information associated with a source document, providing context, descriptions, or annotations. In our ongoing example, we can enrich the image of the Taj Mahal with various textual metadata elements, as presented on the following page. These textual pieces allow the retrieval system to find relevant images with text retrieval methods. For example, if the image's description metadata includes the keywords “Taj” and “Mahal”, we can directly match it with user queries such as “Taj Mahal”.
- However, it is not as easy as it seems. Firstly, we need to gather metadata for the images in the database. How?
  - Manual annotations: human workers inspect each image and provide context, descriptions, categories, tags and other metadata items for the image
  - Automated annotations: technical metadata, such as geo-location, can be captured at the time of taking the image. Additionally, AI workers can analyze an image and extract pre-learned annotations. If the image is embedded in a broader context, such as a web page, that context can yield more information about the image
- Secondly, annotations obtained by two different workers, whether human or AI, can semantically differ from each other disabling a direct matching approach. Let's consider an example:
  - Worker A adds the following keywords: Taj Mahal, India, magnificent building, 17^{th} century
  - Worker B adds the following keywords: religious building, great weather, few people outside, nice

  Both workers have provided accurate annotations, but they differ in semantic levels. When comparing the phrases “Taj Mahal” with “religious building”, a retrieval system must consider the relationships between words. In this case, “Taj Mahal” is a more specific term used by workers familiar with the building, while workers who have never seen the Taj Mahal (or an AI not trained to recognize the building) or lack context may opt for the more generic phrase “religious building”. Similar relationships appear almost everywhere in natural language: horse ↔ mammal, Matterhorn ↔ mountain, Italy ↔ Europe. We will address these relationships and how to handle them later in the course when we study natural language processing and the more recent development of embeddings

  Worker B's use of the keyword “nice” expresses a subjective and abstract concept. Obtaining and normalizing such abstract concepts, where two individuals would agree upon them, can be challenging. However, if we can match these abstract concepts with user preferences, we can provide more relevant examples for the user's queries. For example, if a user plans a visit to India and searches for sites with “great” architecture, both “India” and “great” describe abstract concepts which are not present in the pixels alone, but are obtainable from metadata.

- Let's annotate our ongoing example, the picture of the Taj Mahal. In essence, we can consider three types of metadata: generic, specific, and abstract. Furthermore, we can annotate images across various facets as illustrated below. It's worth noting that we can extract certain metadata like "outside", "time/date taken", "sky", or "building" directly from the raw pixel information, regardless of whether a human or AI worker performs the task. However, other information such as "UNESCO", "Mumtaz", "1648" or even "Taj Mahal" requires a worker who possesses contextual awareness since these details cannot be derived from the pixels alone.



Object Facet	Value
Generic Object Instance	building, water, sky
Generic Object Class	mausoleum, tomb, dome, minaret
Specific Named Object Class	UNESCO World Heritage Site (since 1983)
Specific Named Object Instance	Taj Mahal

Spatial Facet	Value
Generic Location	outside
Specific Location Hierarchy	India, Uttar Pradesh, Agra

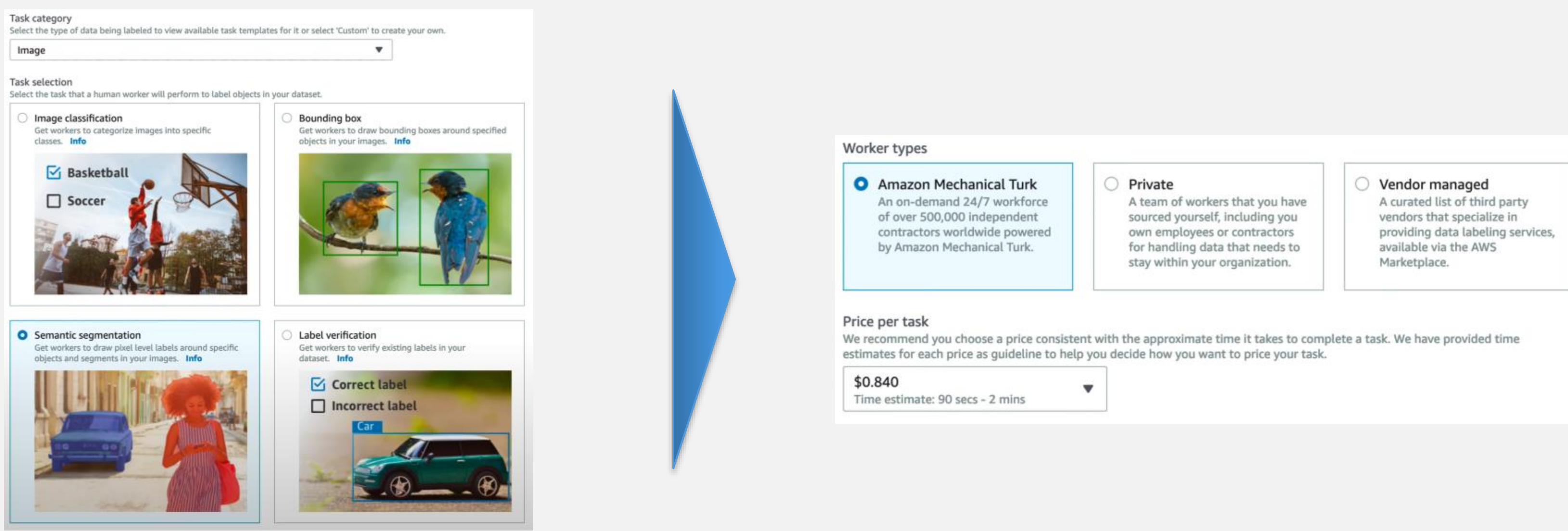
Event / Activity Facet	Value
Generic Event/Activity	tourism, attraction
Specific Event Instance	International World Heritage Expert Meeting on Visual Integrity in 2006

Contextual Facet	Value
Topic	Indian Architecture
Related Concepts / Objects	Shah Jehan, Mumtaz Mahal, Islam
Abstract Concept	love, death, devotion, remembrance
Context	built in memory of his favorite wife Mumtaz Mahal, by Shah Jehan; completed 1648

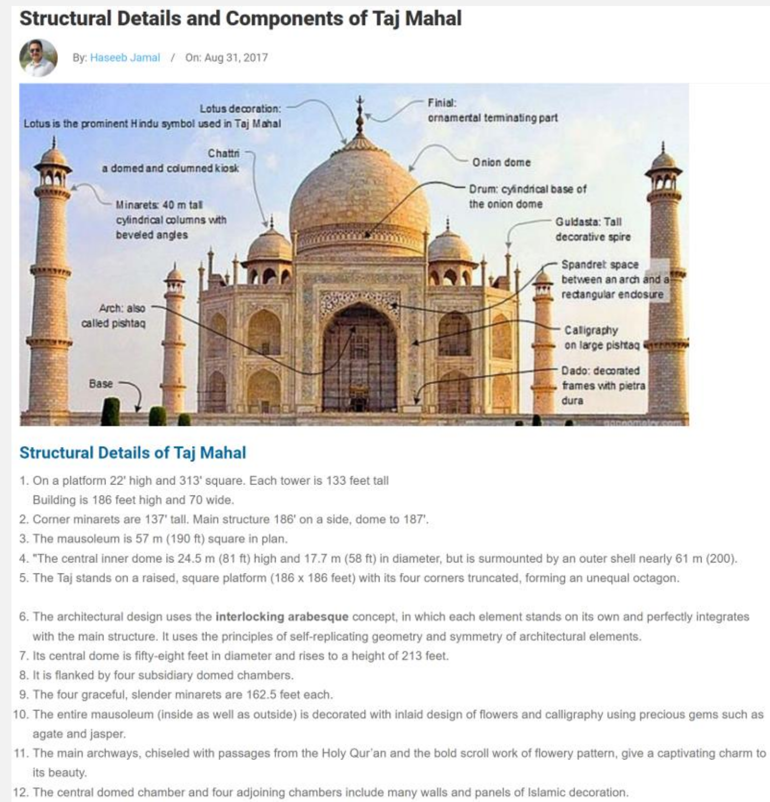
Temporal Facet	Value
Generic Time	summer, daytime
Specific Time	2006 (photo taken)

1.3.1 Manual Metadata Creation

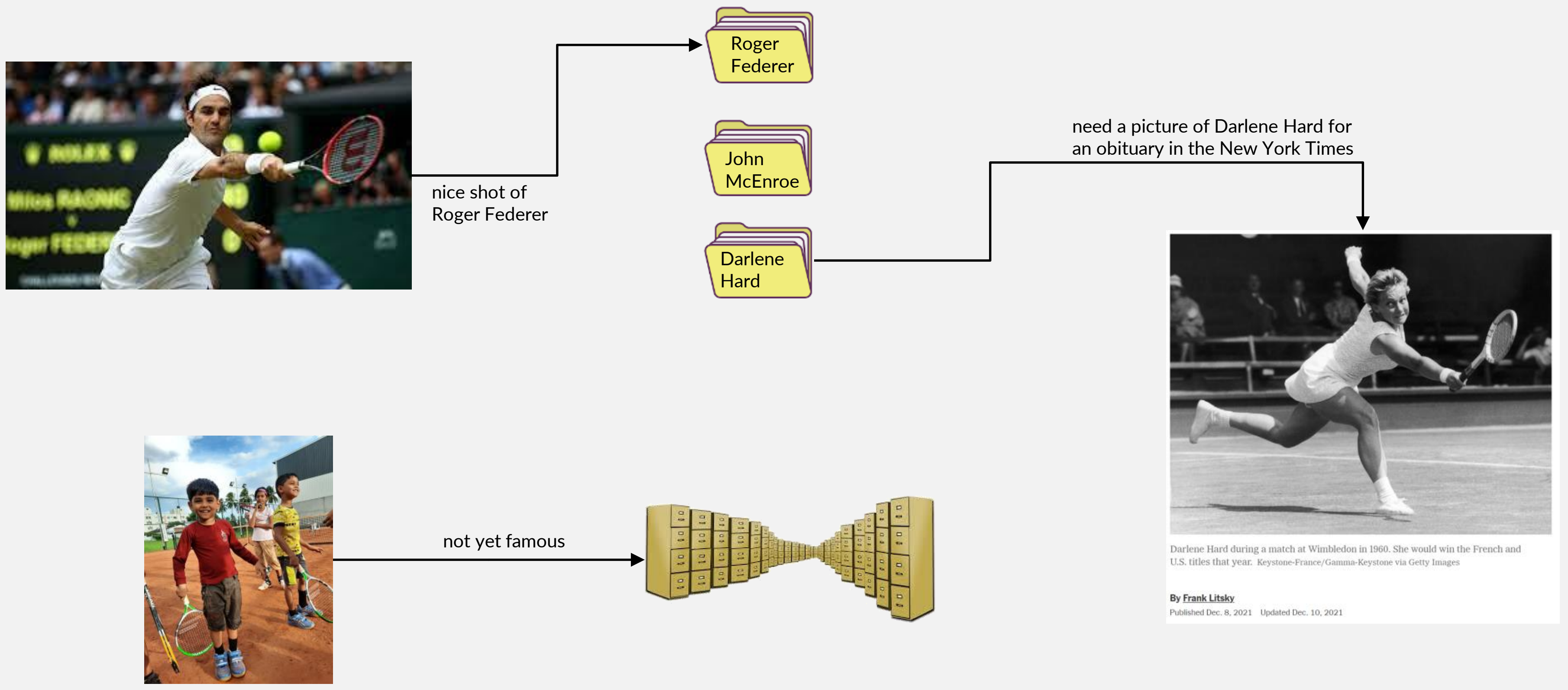
- The process of creating metadata using human workers has gained popularity on various machine learning platforms. In supervised learning, labeled data is required to train models, and these labels at the same time generate metadata for images. For instance, Amazon Mechanical Turk offers access to over 500,000 independent contractors who can perform well-defined tasks at specified prices, as depicted in the example below. Similarly, ChatGPT was trained with the help of thousands of workers to assess the quality of answers generated by the AI.
- Annotation or labeling tasks typically cost around $1 and upwards, depending on the complexity of the task and the required domain knowledge. For basic tasks in machine learning, generic labels and descriptions often suffice. However, annotating stock images or categorizing items in a media archive demands more specific labels and extensive domain knowledge, leading to higher annotation costs. By leveraging a global workforce, annotation tasks can be scaled to millions of items at reasonable costs, yielding results within a reasonable time frame. We will see a few example in one of the upcoming pages.
- In the case of machine learning, the initial investment in training a model can subsequently produce automated labels, as we will explore further in this chapter.



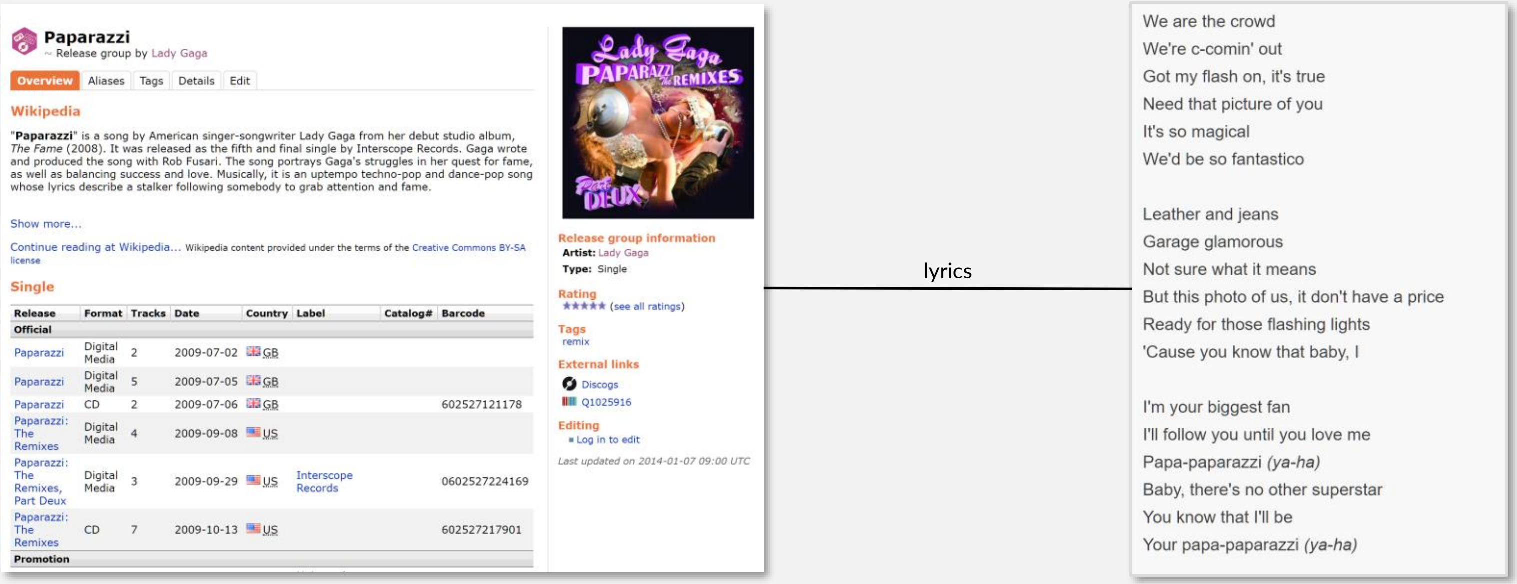
- The quality and substance of manually created labels can greatly vary depending on the domain expertise of the human workers. In the examples provided below, we can observe two distinct approaches to annotating a picture of the Taj Mahal. On the left side, we have the results of a more generic and concise labeling task, whereas the right side shows a comprehensive analysis and description that demonstrates deep domain knowledge.
- This serves as a good example for the challenges when dealing with manually created metadata such as variations in level of detail, choice of keywords, and depth of domain knowledge. The annotations on the left side may not provide sufficient information to boost the image for queries of the Taj Mahal, while the annotations on the right side are so detailed that they are less likely to align with typical queries for the Taj Mahal.



- Stock photo services and media company archives maintain concise keyword lists for each image. They also utilize "faceted navigation" which involves categorizing images based on various attributes with pre-defined values such as prominent individuals, locations, brands, or time periods like decades. For instance, sports event photos are examined to identify shots featuring known individuals. Only a limited number of selected shots from each event are annotated for faceted navigation to keep the overall number manageable. This allows users, like journalists, to easily browse through a curated list instead of scanning thousands of pictures when they need an image of a prominent person. However, one drawback of this approach is that acquiring pictures of individuals before they gain prominence is challenging and often relies on lucky discoveries or contributions from the individuals themselves or their entourage.

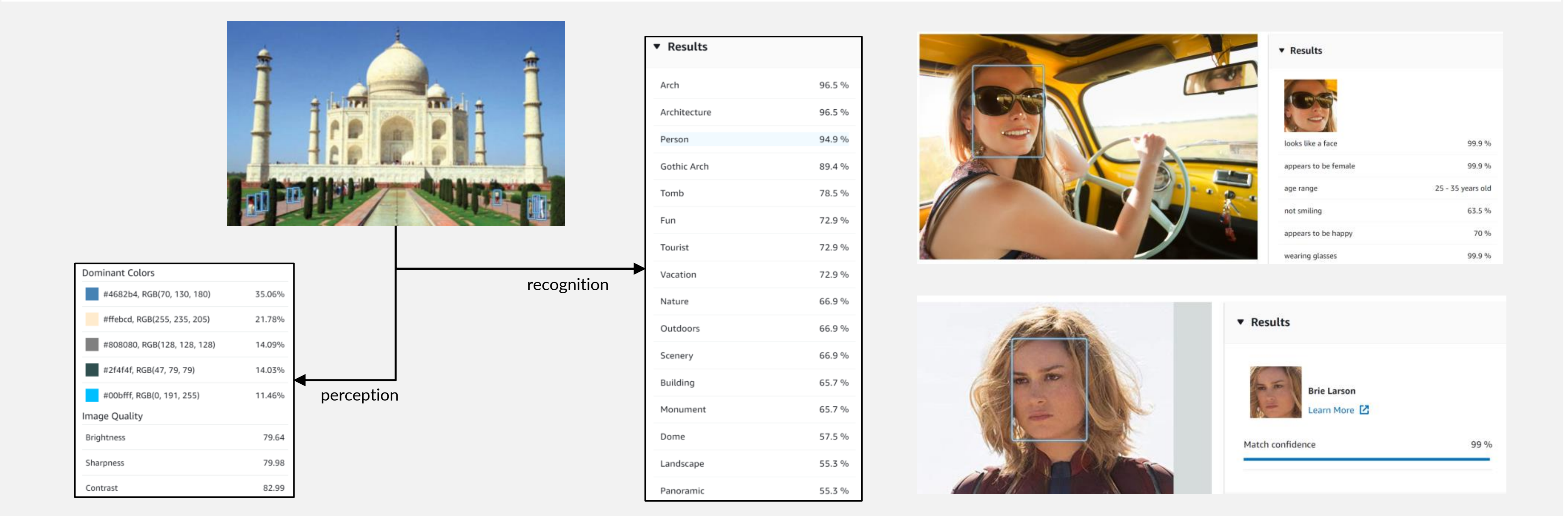


- In domains with a limited set of items, such as songs or movies, metadata annotations with quality control and consistent structure are available. IMDb, a subsidiary of Amazon, is an example of such a database that holds records for movies and episodes from various publishers. Each item is annotated with predefined attributes and has relationships with other items. The database is curated by volunteers, actors, crews, and industry executives, and is accessible online in compiled formats. As such, this is an excellent illustration of “scaled-out” metadata gathering.
- MusicBrainz is another good example for a community-maintained open-source encyclopedia of music information. It provides details about artists, albums, songs, and releases. When combined with a lyrics database, music search benefits from a wide range of textual features and factual data that are not obtainable from the raw audio alone.
- With such curated databases, retrieval of information greatly benefits from high-quality annotations. Often, the metadata alone is sufficient to bridge the semantic gap, meaning that the audio data itself is only used in rare cases, such as with Shazam, to retrieve information about the currently playing song. Due to the commercial and community interest in these domains, the additional efforts involved in creating and maintaining the metadata are covered by increased revenues.

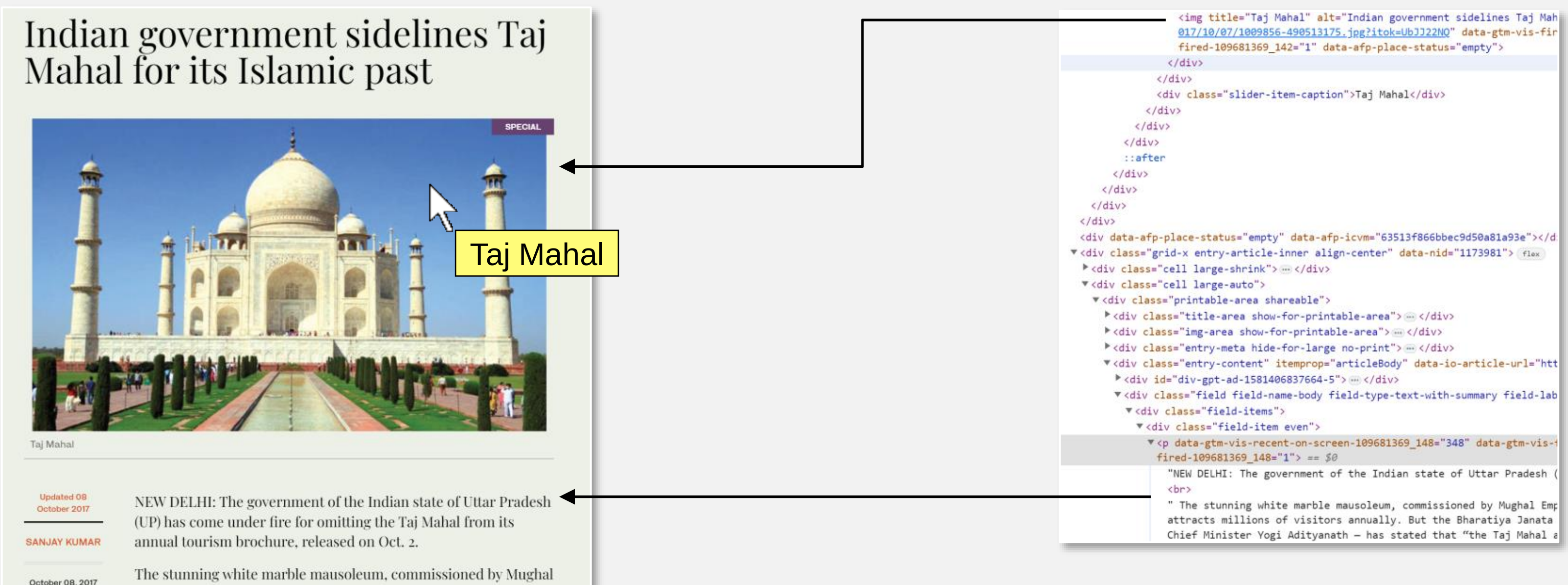


1.3.2 Automated Metadata Extraction

- Annotating arbitrary photos and videos raises challenges due to the absence of a curated reference database for readily obtaining metadata. The costs associated with annotating every single photo and video would far outweigh the added value of having metadata (unless you do it for your photos and videos as fun activity after vacations)
- In such cases, automated annotations and AI-based metadata extraction provide valuable support for retrieval systems. The following examples illustrate the extraction of metadata at different semantic levels:
  - Perception level (left side, lower part): The signal information is processed to capture key aspects that enable comparisons between items based on how humans interpret the signals. This course will provide extensive examples covering various types of multimedia items.
  - Recognition level (middle to right side): Machine learning methods analyze the signal information and its context to extract pre-trained metadata items that can be generic, specific, or abstract. Examples include generic object recognition (architecture, person, female, outdoors), specific object recognition (Brie Larson), or abstract concepts that a typical human would recognize (fun, age, happy).

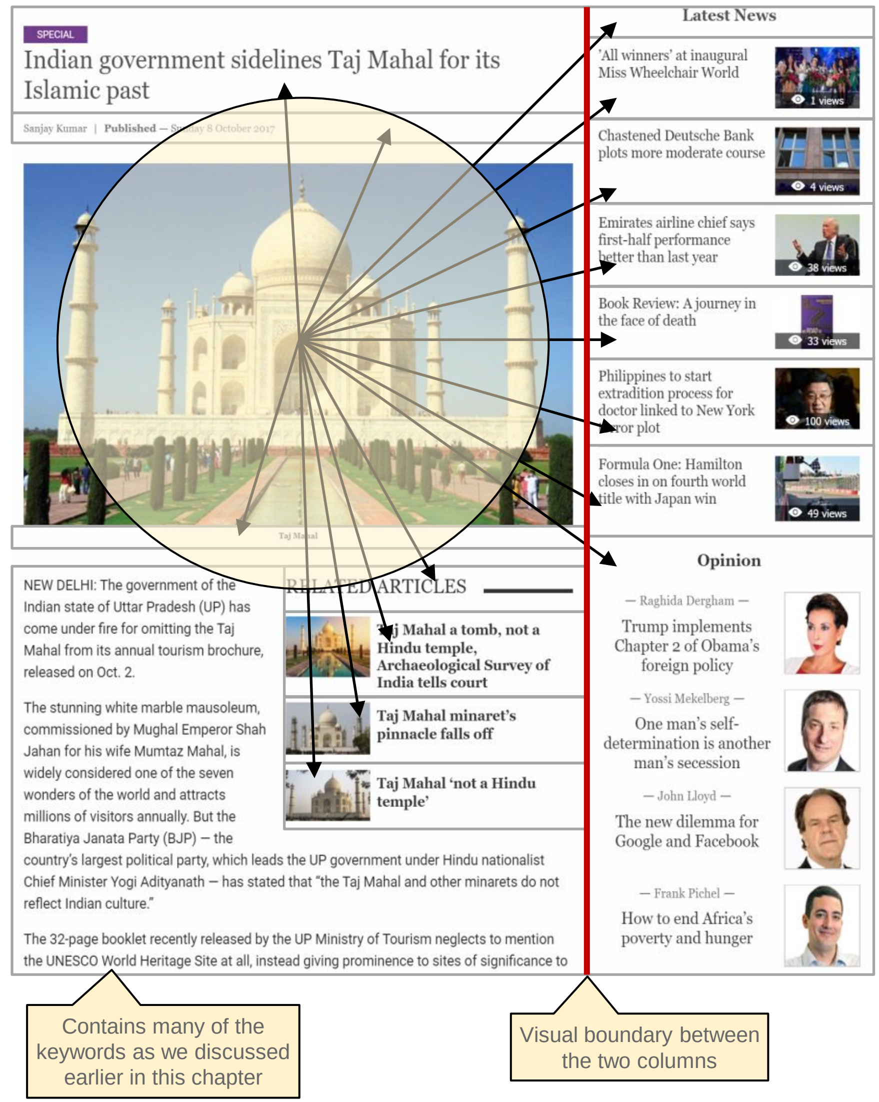


- When documents are embedded on the web, there is a simple yet powerful approach to extracting context, relationship information, and textual metadata:
  - HTML tags such as `<a>` or `<img>` include special attributes that provide descriptions or short annotations for the referenced objects. These attributes can be extracted and used as metadata.
  - The surrounding area on the web page, including title information, text blocks, and captions, can serve as another valuable source of keywords that are likely to overlap with the context of the embedded object. While not always a perfect match, this source often provides valuable information and is easily obtainable.
- In the early days of the web, the "surrounding area" referred to the immediate vicinity within the HTML source code. By considering a window of a few dozen tokens before and after the embedded object, most of the relevant keywords could be captured. Additionally, the header sections (`<h1>`, `<h2>`) and title of the web page were useful sources. However, modern web applications utilize advanced scripts and CSS styles that dynamically change data and layout, making the direct neighborhood within the HTML less reliable for capturing relevant keywords. As shown in the illustration on the right-hand side, the distance between the image and the text paragraph can be large in terms of both text position and hierarchical position due to CSS instructions.

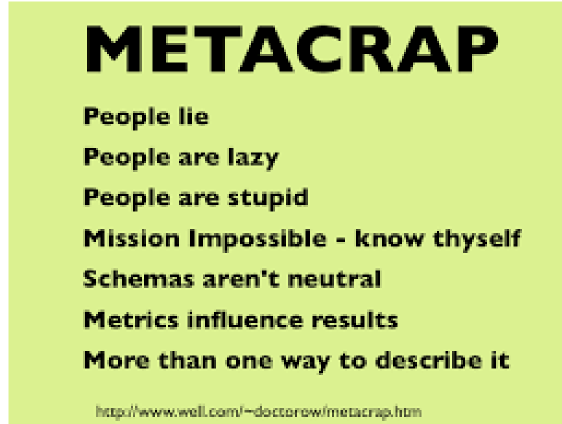


- Advanced web-based metadata extraction considers the visual proximity between embedded objects and text blocks, even though it comes with higher extraction costs. Here's how it works:
  – The web page is rendered in a browser and we identify all objects and text elements of interest
  – Each DOM element has a bounding box, accessible through the `getBoundingClientRect` method which provides on-screen distances between objects
  – We can scan for visual, CSS, or textual cues to eliminate or weigh down text blocks that are not directly relevant such as sidebars or other articles
  – Distances and cues provide proximity weights for the keywords in text blocks that we can use to describe the context of the embedded object

```
function getPositionAtCenter(element) {
    const {top, left, width, height} =
              element.getBoundingClientRect();
    return {
        x: left + width / 2,
        y: top + height / 2
    }
}

function getDistanceBetweenElements(a, b) {
    const apos = getPositionAtCenter(a);
    const bpos = getPositionAtCenter(b);

    return Math.hypot(apos.x - bpos.x,
                      apos.y - bpos.y)
}

getDistanceBetweenElements(image, text)
```



- The trustworthiness of metadata is a subject of concern, as highlighted by Cory Doctorow's "seven insurmountable obstacles" to achieving a meta-utopia. These obstacles include:
  - **People lie**: Unscrupulous content creators may publish misleading or dishonest metadata to redirect traffic
  - **People are lazy**: Many content publishers lack the motivation to thoroughly annotate their published content
  - **People are stupid**: Not all content publishers possess the necessary intelligence to effectively catalog their produced content
  - **Mission impossible—know thyself**: Inadvertently misleading metadata can be published by content creators
  - **Schemas aren't neutral**: Classification schemes are subjective and can introduce biases
  - **Metrics influence results**: Competing metadata standards bodies may never reach an agreement
  - **More than one way to describe it**: Resource description is subjective, and different perspectives exist.



- Absolutely, we should not disregard metadata entirely. Instead, it is important to exercise caution and carefully evaluate the information it provides. High-quality metadata, as seen in platforms like IMDb and MusicBrainz, can be exceptionally valuable. Observational metadata obtained through web crawling can also be beneficial, especially when the system is designed to resist manipulation. For example, the Google web search engine gives higher importance to anchor texts provided by others linking to a page rather than relying solely on the keywords provided by the content owner. However, even these advanced approaches can potentially be manipulated as was successfully demonstrated with the so-called Google-bomb.

1.5 References & Links

- Statista is a reputable online portal for statistics, market research, and business intelligence. **The volumes of data creates world wide**, https://www.statista.com/statistics/871513/worldwide-data-created/

- **The IBM Storage and Information Retrieval System (STAIRS)**, https://en.wikipedia.org/wiki/IBM_STAIRS
Also read the Computerworld article, 1975: https://books.google.com/books?id=X_3_D4RqzvIC&dq=IBM+STAIRS%2FVS&pg=PA14

- Semantic gap: the definition goes much beyond the scope of this introductory example, see Wikipedia
  - A. W. Smeulders, M. M. Worring, A. Gupta, and R. Jain, **Content-Based Image Retrieval at the End of the Early Years**, IEEE Trans. Pattern Anal. Machine Intell., vol.22 no.12, pp1349-1380, 2000. https://doi.org/10.1109%2F34.895972
  - B. Barz, J. Denzler, **Content-based Image Retrieval and the Semantic Gap in the Deep Learning Era**, CBIR workshop at ICPR 2020. https://arxiv.org/abs/2011.06490

- Google Research: Pathways Language Model (PaLM): **Scaling to 540 Billion Parameters for Breakthrough Performance** https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html

- Amazon AWS Machine Learning Blog, **Scaling Large Language Model (LLM) training with Amazon EC2 Trn1 UltraClusters** https://aws.amazon.com/blogs/machine-learning/scaling-large-language-model-llm-training-with-amazon-ec2-trn1-ultraclusters/

- Patrick Lewis et al., **Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks**. 2020, https://doi.org/10.48550/arXiv.2005.11401

- Metadata essay by Cory Doctorow, 2001: **Metacrap: Putting the torch to seven straw-men of the meta-utopia** http://www.well.com/~doctorow/metacrap.htm

- Metadata services, a few examples:
  - Music: AllMusic, Discogs, Last.fm, MusicBrainz, Lyrics.com, Genius
  - Movies/TV: The Movie Database (TMDB), IMDb, AllMovie

- Tom M. Mitchell, **Machine Learning**, 1997, McGraw-Hill Science/Engineering/Math, ISBN: 0070428077

- Ian Goodfellow, Yoshua Bengio, Aaron Courville, **Deep Learning**, 2016, MIT Press, online version: https://www.deeplearningbook.org/

- Various authors, **Dive into Deep Learning**, 2023, to be published at Cambridge University Press, online version: https://d2l.ai/

- **MLOps: Machine Learning Life Cycle**, 2022, https://www.ml4devs.com/articles/mlops-machine-learning-life-cycle/