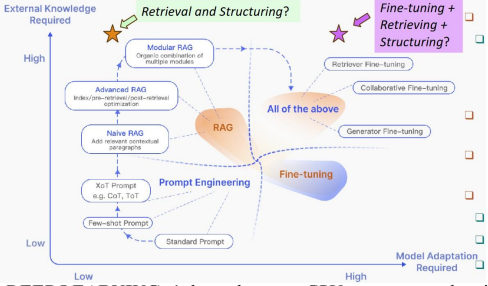


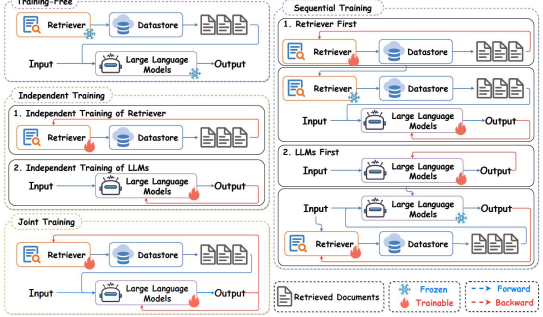
Prompt Engineering: low model modification & external knowledge, harnessing the capabilities of LLMs
Retrieving: Smart Retrieval May Lead to More Effective RAG/LLM
Structured Knowledge may Lead to Better Retrieval for RAG
Structured Knowledge may lead to Better Augmented Reasoning Generation

DEEP LEARNING: 1. large data sets, GPU compute acceleration, advanced neural network structures and tricks: C / RNN, Dropout, ReLU, residual connection
2. **Contextual Embedding:** pretrain-finetune paradigm; (1) Enc: Predict masked tokens based on all others, language understanding / classification tasks (BERT,ELECTRA); (2) Enc-Dec: Generate output seq given masked / corrupted input seq, language understanding and generation tasks (T5, BART); (3) Dec: Next token preds, language generation tasks (GPT, Llama)
3. **BERT:** Embedding + Transformer + FC; MLM + NSP; classifier layer; RoBERTa, ELECTRA (Replaced Token Detection, *discriminator*), COCO-LM (CLM, SCL)
4. **T5, BART:** Mask out spans of texts; generate the original spans; Seq2Seq; masks, delete, permute, denoising autoencoder
5. **GPT:** LMs of billions of parameters with astonishing generalization ability to applications

$$\mathcal{L}_{MLM} = -\sum_{i \in M} \log P(x_i | x_{\setminus M}) \quad \mathcal{L}_{AR} = -\sum_{t=1}^{T} \log P(x_t | x_{<t}) \quad \max_{\theta} \sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t}, \tilde{x})$$

RAG: 1. Hallucination, Outdated, lack of domain-specific kg, expensive updating; look at data, retrieve the important pieces of content, and direct to a large language model as context; accuracy & credibility, continuous knowledge updates, integration of domain-specific info
2. **Retrieval:** raw, chunk (256 - 512), emb, vector db, query. **Sparse**: BM25, word / text retrieval, don't match; **Dense**: vec space, DPR $\text{sim}(q,p) = E_Q(q)^\intercal E_P(p)$, Bi-encoder, BERT; One-encoder: Contriever, Spider, contrastive, versatility
3. **Granularity:** doc, text chunk / passage, token, entity, triplet, subgraph, phrase, sentence
4. **Sources**: Closed-sourced DB (Wikipedia), Indexing (Structural, KG), search engine
4. **Pre-retrieval enhance:** Query expansion, rewrite, augmentation; **Post-retrieval enhance:** RL Distillation, RRerankG, Iter retrieve based on reasoning path, RECOMPress
5. **Input, Output, Intermediate Aug:** Interpolate two next-token distributions in prediction; A transformer module to leverage retrieved info into the generator to interact with the representations in the middle stage (white box)
6. **RAG Fusion:** multi queries of the orig user query, Reciprocal Rank Fusion (RRF), the most relevant doc surface at the top of the search results; various perspectives

RETRIEVAL: 1. **DeepRetrieval**: Retrieval outcome (e.g., recall) is verifiable, reflects the retrieval quality (1) Let LLM learn how to rewrite queries so that retrieval metrics (recall, NDCG) improve, treated as an exploration problem; (2) First performs reasoning <think> to plan how to rewrite the query; (3) The rewritten query is executed by a retriever, producing real retrieved documents; (4) Reward is computed directly from retrieval outcomes, update the LLM using RL; Retrieval First Seq
2. **SearchR1**: (1) train the LLM decide how and when to trigger <search> action to answer questions more accurately; (2) During RL training, The LLM generates an initial response; if it outputs <search>, a search call is triggered. The results from the search engine are injected back into the context as <info>; (3) continues generating reasoning and a final <answer>; (4) Reward = exact match of the final answer, and use PPO/GRPO to update the policy. Joint training
3. **S3**: trains the search agent using Gain Beyond RAG (GBR), a reward that measures how much the retrieved documents improve answer accuracy beyond a naïve RAG baseline, isolates the true contribution of search; Begin with search, LLMs already have strong search capability, and we just need a correct reward and a small amount of data to align them with instructions; RFS
4. **Doc2Q**: Expand documents with the generated queries; **QSetGen**: Identify key topics and phrases in each doc; track which concepts are covered by previously generated queries, increase sampling prob for uncovered concepts and condition the LLM on them to generate new queries
5. **FLARE**: actively checks the model's confidence during generation and triggers retrieval whenever confidence is low, iteratively refining the output with newly retrieved information; **SelfRAG**: Enable the LLM to retrieve on demand, check evidence, and self-critique to produce more grounded outputs; Use reflection tokens (Retrieve / Relevant / Supported / Utility) to explicitly control retrieval and evaluation; Train with SFT on data containing retrieved passages + critique labels; inference: retrieve if needed → generate candidates → self-critique → select the most supported segment; **GraphRAG**: Builds an entity–relationship KG from docs using, clusters the graph into communities and generates summaries for each, uses summaries to answer complex, cross-document queries that naive RAG fails to cover; **HippoRAG**: first constructs a comprehensive KG from all corpus passages in an offline phase. Q arrives, the system extracts its key entities and retrieves associated triples and passages, then uses an LLM as a recognition-memory filter to remove nodes that are topologically connected but semantically irrelevant. Query-related entities are treated as seed nodes and given higher weights, after which Pers PageRank is run to propagate relevance through the graph and surface the most informative nodes and paths. The passages and triples tied to these high-ranking nodes form a focused evidence set that is fed to the LLM; **G-retriever**: answering a query requires finding the relevant part of the G, extracting a compact, query-relevant graph structure that preserves the relational reasoning needed; encodes the Q and computes embedding similarities against all graph nodes and edges to select the most relevant ones, then assembles them into a connected subgraph. This subgraph is textualized into triples or short descriptions and provided—together with the query—to the LLM



StructRAG: Different tasks require different structured represents; Hybrid Structure Router, Scattered Knowledge Structurizer, Structured Knowledge Utilizer, question decomposition, precise knowledge extraction, and final answer inference

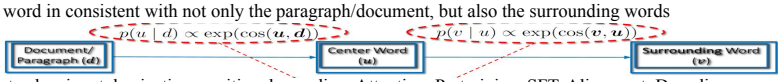
TEXT REPRESENTATION: 1. **Bag of Words, TF-IDF Importance:** Curse of dim + No semantic sim *Term i appears frequently in the current doc d (TF-**log(N(i, d) + 1)**) infrequent in other docs (IDF-**log(N/ni+1)**)*
2. **Distributional Hypothesis:** word characterized by the company it keeps, vector space sim → semantic sim
Text Embedding: One hot → lower dims *Sim (closer) + Linear relations*
3. **Word2Vec:** Max the prob of observing a word based on its local contexts
CBOW (context → word, fast, accurate), Skip-gram (word → context, small. rare) $\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c, j\neq 0} \log p(w_{t+j}|w_t)$
Co-occured words in a **local context window**
4. **Glove:** Solve a least-squares problem to "recover" the co-occurrence matrix
5. **Fasttext:** Shared subword representation; sum of the vector representations of its n-grams
6. **JoSE:** Word Sim = directional (cos) sim, used in application; better clustering performance when normalized

Embedding dot product is optimized during training

$$p(w_O|w_I) = \frac{\exp\left(v'^{\top}_{w_O} v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left(v'^{\top}_{w} v_{w_I}\right)} \qquad J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2 \qquad s(w,c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^{\top} \mathbf{v}_c$$

Word2Vec GloVe fastText

7. **Spherical Generative Model:** general idea of the paragraph/document, and then start to write down each word in consistent with not only the paragraph/document, but also the surrounding words



LLM: 1. data cleaning, tokenization, positional encoding, Attention, Pretraining, SFT, Alignment, Decoding
2. **Data:** Filtering (noise, outliers, imbalance), deduplication
3. **Tokenize:** BytePairEncoding (*Merges the most frequent pairs of characters or subwords to build tokens, keeping common words intact and splitting rare ones*), WordPieceEncoding (*Selects subwords that maximize the likelihood of the training data, reducing unknown tokens through probability-driven merges rather than raw frequency*), SentencePieceEncoding (*Learns subword units directly from raw text without relying on whitespace*)
4. **APE**: Assigns each position a fixed embedding vector independent of the content; **RPE**: Represents the distance between tokens so attention depends on how far apart tokens are rather than their absolute positions; **RoPE:** Use a rotation matrix to encode the absolute position of words and simultaneously includes explicit relative position details in self-attention, captures relative positions in a continuous, flexible way
5. **Pretrain**: Self-supervised; **SFT**: Adjust the model using labeled data so it performs better on a specific task; **Multi-task**: Train the model on multiple tasks to improve its generalization across different types of data and prompts; **Instruction**: Teach the model to follow human-written instructions and respond in ways aligned with human expectations; **Self-Instruct:** Have the model generate its own instruction–input–output examples, filter them, and use them to further improve instruction-following ability; **Prompt-based FT**: Task descriptions are created to convert training examples to cloze questions, MLM
7. **SuperGen** automatically generates class-conditioned training data using a uniD model and then fine-tunes biD on this synthetic data to perform the desired task without manual labels
8. **RLHF**: Train the model to follow human preferences by learning a reward model from human-rated outputs and optimizing the model to produce higher-reward responses; **RLAIF**: Use a stronger, well-aligned AI model to provide preference judgments; **DPO**: Optimize the model directly to prefer "chosen" over "rejected" responses using a simple preference-classification objective, removing the need for a reward model or RL
9. **PPO**: Updates the policy conservatively using a value-function critic to keep learning stable while improving the agent's actions; **GRPO**: Updates the policy by directly comparing multiple sampled actions against each other and boosting the better-than-average ones, without needing a critic
10. **RLHF**: Best when tasks require subjective human preference alignment, because only humans can provide nuanced judgments that models cannot infer themselves; **RLVR**: Best when tasks have objective, verifiable correctness, because the model can optimize without needing human labels.
11. **Decoding**: Greedy, Beam Search, top-k (temperature), Cutoff value-p nucleus (sort, add)
12. **PEFT**: Adapter (bottleneck module), Prefix Tuning (tunable prefix vectors to every Trans layer), LoRA (trainable low-rank matrices to approximate weight updates)
13. **Distillation**: Response, Feature (inter layer, sim rep), API
14. **Limitations**: No memory, stochastic, stale info, training cost, hallucinate (*generated content is nonsensical or unfaithful to the provided source*); structured KG to understand
15. **Prompt:** instructions, questions, input data, and examples, clarity and specificity, crafting the optimal prompt to achieve a specific goal with a generative model (ICL, Reasoning)

REASONING: 1. **KARE:** theme- and task-specific structures, structured-augmented LLM reasoning; integrate high-utility knowledge retrieval & reliable reasoning for interpretable & precise prediction; *(1) Constructs a comprehensive medical concept knowledge graph by integrating information from multiple sources, and performs hierarchical community detection to organize it into a hierarchical community structure; (2) Constructs a base context for patient p, and enrich it with relevant info from the KG and select the most relevant summaries for context augmentation; (3) Generates reasoning chains in a unified format for each patient p and task τ, and fine-tune a relatively small local LLM (7B) to perform reasoning chain generation & label prediction (mortality & Readmission); Accuracy, Macro-F1, Sensitivity, Specificity*
2. **SARG:** flat context, no causal dependencies from unstructured, implicit text, across documents; structure-augmented retrieval for multi-hop reasoning; *(1) Extracts zero-shot causal triple ⟨cause, relation, effect⟩ from documents, perform semantic entity clustering to reduce graph fragmentation & improve reasoning chain connectivity, task-specific knowledge graph; (2) Performs graph-based reasoning by doing semantic node matching to identify the starting point, conducting directed or bidirectional traversal to generate multi-hop reasoning chains, scoring and selecting the most relevant chains as structured reasoning evidence; (3) Augments LLM generation by injecting the selected reasoning chains, together with retrieved text (evidence). into a structured prompt;*
Plug-and-Play, Interpretable reasoning chain, domain adaptability (zero-shot)
3. **Tree-of-Debate:** Explore multi-agent LLM debates to study diverse perspectives and reasoning paths *(1) Converts each paper into a persona and retrieves segment-level evidence from each; dynamically builds a tree of debate topics/subtopics representing contributions and novelties; (2) Each persona self-deliberate (retrieve, prepare arguments), debate (present/rebut), and the moderator determines whether the arguments progressed or introduced any unresolved questions meriting another round; (3) Synthesizes the debate tree into an abstractive comparative summary that highlights similarity, difference, novelty and incremental contributions;*
Knowledge-grounded in a debate elicit critical thinking; structured interactions improve the strength & precision of reasoning; expose models to alternative ideas and critical feedback
Each paper is assigned a persona, and a moderator orchestrates a structured debate. The moderator first asks all personas to state their key contributions, then groups overlapping points into shared subtopics. For each subtopic, the moderator invites only the relevant personas to debate similarities, differences, strengths, and limitations, preventing unnecessary participation and keeping the discussion focused. Additionally, the system can combine pairwise debates as intermediate steps—first generating pairwise comparative summaries among closely related papers, then synthesizing them into group-level analyses. These debates form a larger multi-branch debate tree, with each branch corresponding to a subtopic and containing multi-persona arguments. By combining persona clustering, selective topic routing, and staged pairwise-to-group synthesis, this approach naturally extends Tree-of-Debate to facilitate coherent, fine-grained comparative analysis across multiple scientific papers while maintaining structure, focus, and interpretability.

4. **ClaimSPECT:** break down nuanced claims into specific aspects; *(1) LLM generate a set of coarse-grained aspects for a claim; (2) retrieves and ranks relevant corpus segments to find the most informative and discriminative evidence, expand each aspect into more fine-grained sub-aspects, forming a hierarchy; (3) classifies corpus segments into the appropriate aspect or sub-aspect and finally performs perspective discovery, determining whether the retrieved evidence tends to support, oppose for each aspect, produces a structured, multi-aspect analysis that goes beyond a simple true/false judgment and reveals how different parts of the claim are supported or contested in the corpus*

5. **EpiMine:** identifies fine-grained episodes within large news events; *(1) discovers indicative term pairs that frequently co-occur and signal potential event sub-episodes, partition article segments, separating text into coherent episode candidates; (2) LM refines these partitions by clustering similar segments across top articles and summarizing their shared attributes to form clear episode definitions; (3) all segments are classified into the discovered episodes to produce structured episode clusters that capture the major sub-events within a complex news storyline*

6. **ReasoningBank:** addresses the agent's inability to learn from accumulated interaction history by building a reusable repository of reasoning experience. The method first distills generalizable reasoning strategies from the agent's self-judged successful and failed trajectories, storing these insights as reusable memory items. At test time, the agent retrieves the most relevant memories from this growing experience pool and integrates new learnings back into the repository, enabling continuous self-improvement. The framework further employs MaTTS, a memory-aware test-time scaling mechanism that accelerates and diversifies the learning process by scaling up interaction experience through enhanced retrieval, construction, and consolidation of reasoning traces.

TEXT CLASSIFICATION: 1. Single: Spam Detection; Multi: Paper Topic Classification; Flat: same granularity level; Hierarchical: parent-child relationship, taxonomy

2. Labeling training data is expensive and time-consuming; weakly supervised: a small amount of seed info: category names / keywords (kw level) / few labeled docs (doc level); **Idea**: joint rep learning of words, labels, docs; pseudo labeled training data generate; PLM to classify

3. **WeST:** Embed label names and keywords into the same space, generate pseudo docs from seeds (sample bag of keywords); self training with bootstrapping; **ConWea:** each word find all its occurrences, BERT get context representations, clusters for different senses, extend & expand seed words; **LOTClass:** expands each label's semantics via a BERT (predict what words can replace the label names), generates pseudo-labels for unlabeled texts based on the discovered related words, and iteratively self-trains the model; **ClassKG**: GNN, keyword subgraph, annotate subgraph; **XClass**: clustering for classification based on class-oriented representations, category names only, same corpus could be classified differently; **MEGClass**: word to sentence, sent estimate doc, estimate sent significance using doc, compute doc represent; **PIEClass**: converts the document into the masked token prediction / replaced token detection problem; **CARP**: Collect local fact evidence as clues (keywords, tones), prompt LLMs to go beyond superficial keywords to mine deeper perspectives;

4. **WeSHClass**: tree, one path starting from the root node; **TaxoClass**: relies only on a taxonomy of class names. It first performs a top-down exploration of the taxonomy to narrow down the label search space, using a relevance model (e.g., BERT-NLI) to compute document–class similarity and filter out irrelevant branches. Then, it identifies core classes for each document — those that are both locally (within a taxonomy branch) and globally (across the corpus) most relevant. These core classes serve as high-confidence pseudo labels. Finally, it trains a text matching network between document embeddings and class embeddings, and applies bootstrapping with self-training to iteratively expand labels to more documents, finally achieving large-scale multi-label classification without label or annotation; **TELEClass:** integrates LLMs across the entire weakly supervised classification pipeline. It begins by using an LLM to assign each unlabeled document a small set of initial "core classes" directly from the taxonomy. Then, TELEClass expands the taxonomy by leveraging both the unlabeled corpus and the LLM to generate additional class-indicative terms for each category. Using this enriched taxonomy, the system re-evaluates every document–class pair to compute a relevance score, refining the initial pseudo-labels into more confident core classes. To address data scarcity, TELEClass performs path-guided data augmentation, prompting the LLM to generate synthetic documents for specific taxonomy paths (e.g., creating examples about "hair care → conditioner"). Finally, a multi-label classifier is trained on both real and augmented data, where a document encoder and class-embedding network learn to match documents with their most relevant classes. Overall, TELEClass combines LLM semantic reasoning with taxonomy structure to enable scalable, zero-shot hierarchical text classification without manual labeling

5. Text Classification can help structure the retrieved data, IE, KG construction

STRUCTURE MINING: 1. FET, **OntoType:** generates candidate types using head-word parsing and masked language model (MLM) prompting to infer plausible categories from context. Next, it aligns generated candidate types to several high-level types in the type ontology by Word2Vec+ cosine similarity, selects the most accurate high-level types with NLI. Finally, it progressively refines type resolution, applies NLI entailment model to match each candidate with the appropriate node in a predefined ontology, moving from coarse-grained to fine-grained types to produce accurate, context-aware type assignments

2. **OnEFET:** Enrich the ontology with instance and topic, generate pseudo training data, train NLI model for coarse-to-fine typing; RAG: less fine-grained information, low information density; **ORAG:** Ontology-guided info. propagation, kg-unit, knowledge-dense retrieval

3. **RolePred**: Infer a set of argument role names for a given event type to describe the crucial relations between the event type and its arguments (open-vocab); Predict candidate role names for named entities by casting it as a prompt-based in-filling task, Context. According to this, the ⟨MASK SPAN⟩ of this Event Type is Entity; Formulate the argument extraction problem into question-answering task; Role Filtering by judge the salience of an argument role by involving multiple event instances of the same type & Merging different roles with similar semantics and share the same arguments in an event

4. **ReactionMiner:** automatically extracts structured chemical reaction information such as reactants, products, and experimental conditions from scientific papers. It first converts PDFs to raw text, segments text into localized reaction descriptions, and then enriches predefined reaction roles using GPT-4–based RolePred to discover new roles and synthesize training data. Finally, a unified extraction model is trained to integrate these enriched roles, enabling accurate, fine-grained reaction information extraction that outperforms GPT-4 in quality; **ReactIE:** performs weakly supervised chemical reaction extraction by combining linguistic pattern mining and domain-specific rules to automatically label reaction data. It builds synthetic QA-style training data from both literature and patents, enabling models to extract structured reaction information without manual annotation.; **ActionIE**: treats scientific text as executable instructions, rephrases it for clarity, and then uses LLMs to extract and formalize experimental actions into structured, code-like representations for automated reasoning and reproducibility.

5. **ThemeKG, PrioRE**: Theme-based entity ontology from Wiki, Construct a relation ontology

PrioE: introduces a two-stage process for theme-specific open relation extraction. In the A Priori Seed Generation stage, it first collects entity type hierarchies relevant to a given theme (e.g., Battery → Rechargeable Battery, Vehicle → Electric Vehicle) to build a structural backbone. Then, it leverages an LLM to generate potential relation templates, such as be power source of or be managed by, and compiles them into a Dynamic Relation Dictionary that serves as prior knowledge. In the A Posteriori Seed-Guided Relation Extraction stage, the model identifies topic-relevant entities from documents and locates them within the hierarchy, using the seed relations to detect connections between entities (e.g., (deep cycle battery, be power source of, forklift)). If no suitable relation exists, the LLM infers a new one and updates the dictionary, allowing the system to continually expand its relation space. Intuitively, this approach starts from a small set of structured "seeds" and lets the model grow a comprehensive, theme-aware knowledge graph by iteratively discovering new, meaningful relations.

TOPIC MINING: 1. **TopClus:** Topics are interpreted via clusters of semantically coherent and meaningful words, pretrained embedding spaces do not exhibit clearly separated clusters, mismatch between the number of clusters in the pre-trained LM embedding space and the number of topics to be discovered; perform topic discovery by first projecting PLM embeddings into a lower-d spherical latent space that is explicitly structured for clustering. In the latent space, the model assumes K latent topic directions, and uses an EM-style training procedure to jointly optimize three objectives: preserving the semantic geometry of the original PLM embeddings, ensuring that the learned topics can meaningfully summarize documents, and enforcing clear, well-separated cluster structures; Latent space: low-d, spherical, discard information that is not helpful for forming topic clusters

2. **CatE**: leverages category names to learn word embeddings with discriminative power over the specific set of categories; assumes a simple text-generation process in which a document reflects a category, each word reflects the document's semantics, and local context reflects a word's meaning. CatE jointly learns word embeddings that align with category names, preserve semantic relations from corpus contexts, and capture how specific or general each word is. After training, CatE retrieves representative words for each category by selecting terms that (1) have high cosine similarity to the category name, (2) possess sufficient semantic specificity, and (3) have not been previously selected. The result is a discriminative, category-separable embedding space where each category is associated with a coherent set of key representative words. CatE differs from Word2Vec because it uses category names as explicit semantic anchors, combining document-level signals and word specificity to learn category-separable embeddings rather than only local-context semantics. Compared with TopClus, CatE is category-guided while TopClus is purely unsupervised clustering based on PLM embeddings. Discriminative topic mining yields higher-quality topic phrases because the category names and specificity constraints directly push the model to select focused, meaningful, and non-generic representative terms instead of relying on unguided statistical patterns like traditional topic models.

3. **SeedTopicMine**: integrates multiple complementary context signals to perform seed-guided topic discovery. Given user-provided seed words and a corpus, it first constructs seed-guided text embeddings and extracts PLM (e.g., BERT) representations to produce an initial ranking of candidate topic terms. Using these ranked terms, the method retrieves topic-indicative sentences by identifying anchor sentences that contain only one category's indicative words and expanding them with clean neighboring sentences. Finally, SeedTopicMine combines the three context sources through an ensemble ranking procedure to refine each category's word list. This yields more accurate, purer, and more discriminative topic word sets than relying on any single context signal alone.

RETRIEVAL++: 1. **ToTER**: ToTER builds a taxonomy-guided semantic index for each document by assigning both high-level core topics (derived from a corpus-level topical taxonomy) and fine-grained indicative phrases (identified via phrase mining). This structured index captures both the latent topic hierarchy and the key concepts of each document. Trains an index network that learns to predict topics and phrases from text, and integrates these signals into a base retriever through contrastive fine-tuning with topic-aware negative mining. Combined with the earlier components—taxonomy-guided topic relevance learning, search-space adjustment, class-relevance matching, and query enrichment—ToTER uses the taxonomy and the learned semantic index to guide retrieval, improve relevance, and enhance PLM-based retrievers in a plug-and-play manner

2. **PairSem**: representing relevant kg as entity–aspect pairs, capturing complex, multi-faceted relationships, convert the user query into (entity, aspect) pairs, and perform pairwise semantic matching between query pairs and document pairs, improve scientific document retrieval by matching precise semantic relations

3. **Hypercube**: assumes that queries and documents can be described along several predefined semantic dimensions (e.g., location, event, theme), and it indexes the corpus into a multi-dimensional hypercube where each axis corresponds to one such dimension. LLM decomposes the query into its key dimensions, and retrieval is performed by selecting documents from the cube cells that match these components. The system ranks documents by coverage completeness—prioritizing those that satisfy all query dimensions, and falling back to those with the highest partial coverage when full matches are unavailable

5. **Multicube**: Split the complex query into simple subqueries; Pick the best cube for each subquery to retrieve needed facts; Use retrieved answers to generate the next subquery in an iterative loop; Stop when enough context is gathered, then produce the final answer

6. **PropRAG**: Shift from triples to context-rich propositions and introduce an efficient, LLM-free online beam search over proposition paths to discover multistep reasoning chains

LLM ++: 1. **CoT**: Simulates human-like reasoning processes by delineating complex tasks into a sequence of logical steps (intermediate thoughts) towards a final resolution, 0-shot / Manual eg; **SC:** generate multiple responses to the same query, the consistency among these serves as indicator of accuracy and reliability; **ToT**: Consider various alternative solutions or thought processes, branch out into thought trees (real time analysis and comparison), before converging on the most plausible one; Expert Prompting: role playing; **Rails**: control the output of LLMs through predefined rules or templates

2. **ToolFormer**: decide what tool to use when; **Agent**: A system based on a specialized instantiation of an (augmented) LLM that is capable of performing specific tasks autonomously, Core, Memory, Tool, Planning;

Answer format: The proposed method addresses the problem of XXX by following a multi-stage framework that transforms raw input into structured, interpretable output. It first performs XXX to identify or extract the core elements relevant to the task, producing XXX as the initial representation. Based on this, the system applies XXX—such as retrieval, ranking, clustering, or segmentation—to organize and refine the information and to retain the most relevant or discriminative components. Finally, the method performs XXX, for example structured reasoning, comparison, debate, or synthesis, to integrate the processed information and generate a coherent, task-specific output. Overall, the approach combines XXX, XXX, and XXX to achieve a more reliable and explainable solution to the problem.

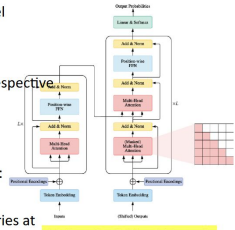
Attention Modules: Transformer adopts the Query-Key-Value (QKV) model

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D_k}}\right)\mathbf{V} = \mathbf{A}\mathbf{V}$$

- Transformer uses multi-head attention:
  - encoder self-attention
  - decoder masked self-attention
  - encoder–decoder cross-attention
- Dm-dimensional original queries, keys and values are projected into respective dimensions, with H different sets of learned projections

$$\text{MultiHeadAttn}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Concat}(\text{head}_1,\cdots,\text{head}_H)\mathbf{W}^O,$$
$$\text{where head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V).$$

- Three types of attention based on source of queries and key-value pairs:
  - Self-attention: In encoder, set Q=K=V=X (outputs of the previous layer)
  - Masked self-attention: In decoder, the self-attention is restricted: queries at each position p can only attend to all key-value pairs up to and including p
  - Cross-attention: Queries are projected from the outputs of the previous (decoder) layer. Keys and values are projected using the outputs of the encoder



Ack. Figures from T. LIN, et al. A Survey of Transformers, https://arxiv.org/abs/2106.04554v2, 2021

Position-wise FFN: A fully connected FFN that operates separately and identically on each position

Position encodings: Additional positional representation is needed to model the ordering of tokens