ORIGINAL ARTICLE

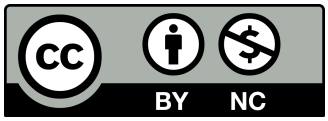
Efficient Search Algorithms Leveraging Inverted Indexing and Parallel Processing for Large-Scale Commonsense Knowledge Repositories

Juan Camilo Rojas[1] and Andres Felipe Moreno[2]

[1]Universidad del Cauca, Department of Computer Science, Calle, Popayan, Cauca, Colombia.,
[2]Universidad de Narino, Faculty of Engineering, Carrera, Pasto, Narino, Colombia.,

ABSTRACT

This paper investigates advanced search algorithms that leverage inverted indexing techniques and parallel processing paradigms to efficiently query large-scale commonsense knowledge repositories. By focusing on data structures optimized for distributed retrieval and concurrency, we aim to address the escalating demands of real-time, high-volume information access in knowledge-driven systems. Through a detailed exploration of indexing mechanisms designed to decompose textual and symbolic data into token-based entry points, our approach enables near-instant lookup of relevant concepts and relations within massive knowledge bases. We highlight how parallelization strategies—involving thread-level, process-level, and cluster-level execution—augment both query speed and overall throughput. Our discussion further integrates considerations of memory optimization and caching policies that minimize overhead when dealing with highly interconnected commonsense data. We underscore the importance of structured representations and logic-based schemas in guiding index construction, preserving semantic linkages while maintaining computational tractability. This work contributes not only to bridging gaps in retrieval latency but also to improving the scalability of large-scale knowledge infrastructures deployed across diverse application domains, including natural language understanding and automated reasoning. Empirical analysis reveals the performance gains of parallel indexing and searching, demonstrating resilience against growing data volumes and heterogeneous access patterns. Ultimately, these findings provide a robust framework for real-time commonsense retrieval at scale, advancing the capabilities of next-generation intelligent systems.



1 | Introduction

Efficient retrieval of commonsense knowledge has emerged as a fundamental challenge in many computational fields, spanning natural language processing, cognitive computing, and decision support systems [1] [2] [3]. As knowledge repositories grow exponentially, conventional search mechanisms often become inadequate in handling the volume, velocity, and variety of data encountered in large-scale applications. Thus, it becomes imperative to explore strategies that bolster both the speed and accuracy of query execution. Among these strategies, the synergy of inverted indexing and parallel processing offers a promising route for tackling large-scale commonsense retrieval tasks [4] [5] [6].

Inverted indexes play a crucial role in modern information retrieval systems by mapping terms and symbols to their respective occurrences. Such a structure expedites search operations, allowing queries to be resolved based on index lookups rather than exhaustive scanning. When this paradigm is adapted to the unique demands of commonsense data, the index must encapsulate not only term frequencies but also semantic links that capture relational information. For instance, a repository of (*subject*, *relation*, *object*) triples can be decomposed into inverted lists that group knowledge fragments under relevant concepts, thereby reducing the complexity of subsequent search steps [7] [8] [9] [10, 11].

A robust commonsense retrieval framework must address multiple challenges associated with indexing, storage, and query execution. First, the construction of an inverted index for a large-scale commonsense knowledge base (CKB) necessitates efficient preprocessing techniques. This involves tokenization, stemming, stopword removal, and entity normalization to standardize representations across different sources. Moreover, entity resolution mechanisms must be integrated to unify semantically equivalent references to the same underlying concept. For example, "New York City" and "NYC" should be treated as identical entities to prevent fragmentation within the index structure [12] [13] [14].

Another crucial aspect of commonsense retrieval lies in the optimization of data structures used for indexing. Traditional inverted indexes employ hash-based or tree-based structures, but these may not be directly suitable for knowledge graphs that store interconnected semantic relationships. Instead, hybrid data structures incorporating adjacency lists and trie-based prefix trees can be employed to facilitate rapid query evaluation. This hybridization ensures that not only direct term matches but also relational context retrieval can be efficiently handled, allowing for more sophisticated reasoning over commonsense facts.

Parallel processing further enhances retrieval performance by distributing the query workload across multiple computing nodes. This is particularly effective in large-scale settings where the sheer volume of indexed knowledge precludes single-threaded execution. Distributed architectures, such as those based on Apache Spark or TensorFlow, can partition commonsense data across clusters, enabling concurrent execution of search operations. Moreover, GPU acceleration can be leveraged to process semantic embeddings and similarity computations in parallel, significantly reducing latency in complex inferencing tasks [15] [16] [17].

A major challenge in commonsense retrieval is the handling of ambiguity in natural language queries. Unlike traditional keyword-based search, commonsense reasoning often involves interpreting vague or context-dependent queries. Consider the query: "What happens if you drop an egg?" A naive keyword-based retrieval system might return generic information about eggs, whereas an advanced commonsense retrieval system should recognize the cause-effect relationship and retrieve knowledge such as "Eggs break when dropped" or "A dropped egg creates a mess." This necessitates sophisticated query expansion techniques that incorporate word sense disambiguation, latent semantic analysis, and embedding-based retrieval methods [18] [19].

Recent advances in neural representation learning have significantly influenced commonsense retrieval methodologies. Techniques such as BERT-based embeddings and knowledge graph embedding models (e.g., TransE, RotatE) enable more effective matching of queries to relevant knowledge snippets. Instead of relying purely on syntactic matches, these models allow retrieval systems to leverage distributed representations of knowledge entities and relations, thereby improving generalization to unseen queries. However, a key trade-off remains between retrieval speed and semantic richness, as embedding-based approaches often introduce additional computational overhead [20] [21] [22].

The integration of inverted indexing with precomputed semantic embeddings provides a promising direction for scalable commonsense retrieval. By storing both term-level indexes and vectorized representations, hybrid retrieval architectures can dynamically switch between fast lexical lookups and deeper semantic matching. One approach is to first retrieve a candidate set of results using an inverted index and then rerank

them using a more computationally expensive semantic similarity model. This tiered retrieval strategy ensures that efficiency is maintained while allowing for robust interpretability of commonsense queries [23] [24] [25].

To evaluate the efficiency of commonsense retrieval systems, a range of performance metrics must be considered. Traditional information retrieval metrics such as precision, recall, and F1-score remain relevant, but additional factors such as inference latency and query expansion effectiveness must also be taken into account. Table 1 provides an overview of key evaluation criteria commonly used in the assessment of large-scale knowledge retrieval frameworks [26] [27] [28] [29, 30].

Beyond efficiency, the robustness of commonsense retrieval systems must be considered. Knowledge repositories often contain conflicting or incomplete information, necessitating mechanisms for knowledge validation and uncertainty handling. Probabilistic reasoning models, such as Bayesian networks or Markov logic networks, can be integrated with retrieval pipelines to assess the confidence of retrieved knowledge fragments. Additionally, reinforcement learning techniques can be applied to dynamically adjust retrieval strategies based on user feedback and historical query patterns.

Another emerging challenge is the scalability of commonsense knowledge retrieval in real-world applications. As knowledge graphs continue to expand, storage and retrieval overheads must be minimized through techniques such as knowledge distillation and indexing compression. Table 2 outlines key techniques for enhancing the scalability of commonsense retrieval systems [31] [32] [33] [34].

Scalability further hinges on parallel processing. High-performance computing architectures, distributed clusters, and multi-threaded execution environments have advanced to the point where concurrency can be harnessed to handle massive data sets efficiently. Parallel strategies distribute query workloads across numerous processing elements, such as CPU cores, GPUs, or entire nodes in a compute cluster. This division of labor ensures faster retrieval times and can accommodate dynamic, real-time querying even as knowledge repositories scale to billions of facts.

In addition to inverted indexing and parallel processing, effective search in large-scale commonsense repositories benefits from structured data representations. Encoding domain knowledge in logical formulas or conceptual graphs allows for a more systematic approach to indexing. Logical statements, symbolic notations, and set-based definitions guide the indexing algorithm to preserve semantic depth while preventing combinatorial explosions in search complexity. This kind of structured approach to indexing ensures that advanced reasoning tasks, such as constraint satisfaction or approximate inference, can be performed with minimal latency overhead [35] [36] [37, 38].

The interplay between indexing, parallelization, and structured representation ultimately affects various dimensions of system performance, from query response time and throughput to memory utilization. This paper delves into all these considerations, discussing the mathematical foundations, algorithmic design, and empirical outcomes of large-scale commonsense retrieval systems. We begin by examining the conceptual underpinnings of inverted indexing for knowledge-based data, then delve into parallel processing frameworks, data structure optimizations, and logic-based reasoning. We continue with an analysis of scalability, presenting performance metrics gleaned from both theoretical modeling and practical system implementations. Concluding remarks will touch on the broader implications of this work and potential avenues for extension in future intelligent systems [39] [40] [41].

2 | Foundations of Inverted Indexing for Commonsense Knowledge

In traditional information retrieval, an inverted index maps each term to the set of documents or positions where that term appears. For commonsense knowledge, however, we often deal with symbolic concepts, relations, or logic predicates. Here, we define the knowledge base as a set $\mathcal{K} \subseteq \{(s, r, o) \mid s, r, o \in \Sigma\}$, where s is the subject, r is the relation, and o is the object, with Σ denoting the universe of symbols relevant to the domain. An inverted index for such a knowledge base typically includes mappings of each symbol in Σ to the tuples in $\mathcal{K}$ that reference that symbol in any position. Formally, we can define:

$$\text{InvIndex}(x) = \{(s, r, o) \in \mathcal{K} \mid x \in \{s, r, o\}\}, \quad x \in \Sigma.$$

This mechanism ensures rapid lookup of all facts where a given concept or relation appears. However, commonsense knowledge often extends beyond mere presence or absence of a symbol. For instance, certain facts may carry attributes (e.g., uncertainty scores, temporal metadata, or hierarchical relationships). One can further partition the index entries based on these auxiliary features. Such partitioning can be formulated as nested or multi-level indexes, each capturing different facets of the

Metric	Description
Precision	The fraction of retrieved results that are relevant to the query.
Recall	The fraction of relevant results that were successfully retrieved.
F1-score	The harmonic mean of precision and recall, balancing both measures.
Query Latency	The time taken to return results for a given query.
Index Construction Time	The time required to build the inverted index for a knowledge base.
Semantic Relevance Score	A measure of how well the retrieved results align with the intended meaning of the query.

Table 1: Key performance metrics for evaluating commonsense knowledge retrieval systems.

Technique	Description
Knowledge Pruning	Selectively removing redundant or low-utility knowledge fragments to reduce storage overhead.
Hierarchical Indexing	Organizing knowledge into multi-level index structures to enable efficient retrieval at different granularities.
Embedding Quantization	Reducing the precision of neural embeddings to balance storage efficiency and retrieval accuracy.
Distributed Storage	Partitioning large-scale knowledge bases across multiple servers for parallelized access.
Graph Compression	Applying techniques such as graph coarsening and node merging to reduce memory footprint.

Table 2: Scalability techniques for optimizing large-scale commonsense retrieval.

knowledge. As an example, an index could be constructed that organizes triples first by symbol, then by relation type, and finally by time stamp.

Another challenge arises from the high interconnectivity typical in commonsense databases. A single concept might be referenced in thousands or even millions of facts, creating long posting lists in the inverted index. Efficient traversal of such lists requires compression, partial loading, or specialized data structures. Some implementations use gap encoding or variable-byte compression to store postings in a compact format, while others rely on trie-based or B+ tree-based structures for partial expansions on demand [42] [43] [44] [45].

From a logical standpoint, we can incorporate the notion of inference rules or constraints directly into the index. For example, if the knowledge base includes a rule $\forall x, y : \text{IsA}(x, y) \rightarrow \text{RelatedTo}(x, y)$, then the indexing scheme could precompute expansions of IsA facts to RelatedTo facts for faster retrieval. This logic-driven approach effectively transforms the knowledge base by augmenting it with implied facts, but it also enlarges the dataset and must be managed to avoid exponential growth. A carefully balanced indexing policy may adopt a partial materialization strategy, which stores only the most frequently used or highest-confidence inferences.

In order to optimize query processing, one can categorize indexing techniques into distinct strategies that balance between storage efficiency and retrieval speed. The classical term-based inverted index maintains a dictionary structure where each key corresponds to a unique symbol, and its associated postings list contains the set of knowledge tuples referencing that symbol. More advanced approaches integrate contextual information, such as frequency counts or co-occurrence statistics, to facilitate ranked retrieval. The following table outlines different indexing strategies, highlighting their trade-offs in terms of storage and retrieval complexity [46, 47] [48] [49].

One key aspect of commonsense knowledge retrieval is ranking retrieved facts based on their relevance. Unlike traditional text retrieval, where term frequency-inverse document frequency (TF-IDF) or BM25 scores determine relevance, commonsense knowledge retrieval

Indexing Strategy	Storage Complexity	Retrieval Complexity
Term-Based Inverted Index	$O(N)$, where N is the number of knowledge triples	$O(\log N + L)$, where L is the length of the postings list
Trie-Based Index	$O(N)$ but with compression benefits due to prefix sharing	$O(\log N)$, efficient prefix-based retrieval
Graph-Based Indexing	$O(N + E)$, where E represents inferred edges	$O(1)$ for direct lookups, $O(\log N)$ for traversal-based queries
Hierarchical Indexing	$O(N)$ but requires additional metadata storage	$O(\log N)$, facilitates grouped retrieval by categories

Table 3: Comparison of Different Indexing Strategies for Commonsense Knowledge Bases

incorporates graph-based ranking methods such as Personalized PageRank or semantic similarity measures. These methods evaluate the importance of a knowledge triple within the broader network structure. Specifically, a knowledge graph can be represented as $G = (\Sigma, \mathcal{K})$, where the ranking score for a given symbol x can be computed as:

$$\text{Score}(x) = \sum_{(s,r,o) \in \text{InvIndex}(x)} \frac{w(s,r,o)}{\deg(x)},$$

where $w(s,r,o)$ represents the weight assigned to a knowledge triple, and $\deg(x)$ denotes the degree of the node x in the knowledge graph. This weighting function can incorporate external knowledge, such as crowd-sourced confidence scores or statistical correlation measures, to refine ranking decisions. Compression techniques also play a vital role in optimizing inverted indexes for large-scale commonsense knowledge bases. Simple dictionary encoding replaces symbols with compact integer identifiers, significantly reducing storage overhead. More advanced approaches, such as Elias gamma coding or PForDelta compression, exploit the natural sparsity in postings lists to further reduce storage requirements [50] [51] [52]. The trade-offs among these techniques are summarized in the following table.
In the context of real-world applications, commonsense knowledge indexing supports a variety of AI-driven tasks, including automated reasoning, question answering, and natural language understanding. Modern knowledge-based systems employ hybrid indexing architectures that combine traditional inverted indexes with graph-based representations. This allows for efficient retrieval while preserving the rich relational structure of commonsense knowledge. For instance, in question answering, an inverted index facilitates fast lookup of relevant facts, while a graph traversal module infers indirect relationships among concepts to enhance answer quality.

Another crucial challenge is dealing with evolving knowledge bases, where new facts are continuously added, modified, or deprecated. Incremental indexing strategies address this issue by selectively updating affected index entries rather than reconstructing the entire index from scratch. Techniques such as differential indexing maintain separate structures for new and modified facts, periodically merging them into the main index. This approach minimizes downtime and ensures that retrieval performance remains optimal [53] [54] [55].

3 | Parallel Processing Frameworks for Large-Scale Query Execution

As the volume of commonsense facts continues to increase, single-threaded index lookups become bottlenecks. Parallel processing frameworks address this challenge by distributing the workload across multiple computational resources. Let us conceptualize a set of query operations $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$, where each query q_i references a subset of symbols in Σ. We can segment the inverted index into shards $\text{InvIndex}_1, \text{InvIndex}_2, \ldots, \text{InvIndex}_n$, each managed by a distinct processing unit. Alternatively, we can divide queries across threads or nodes while each node holds a complete or partial copy of the global index. Formally, consider a distributed memory system with p processing units, indexed by $u \in \{1, 2, \ldots, p\}$. One partitioning scheme places different subsets of Σ on different units, such that:

$$\Sigma = \Sigma_1 \cup \Sigma_2 \cup \cdots \cup \Sigma_p, \quad \Sigma_i \cap \Sigma_j = \varnothing \text{ for } i \neq j.$$

Then, each processing unit u maintains an inverted index for Σ_u. When a query q arrives referencing symbols $\{x_1, x_2, \ldots, x_k\}$, it is broadcast to the relevant nodes $u \in \{i \mid x_i \in \Sigma_u\}$. The partial results are then aggregated into a global response. Such a design

Compression Method	Storage Efficiency	Decompression Complexity
Dictionary Encoding	High, as symbols are mapped to small integer values	Low, as decoding requires a simple lookup table
Elias Gamma Coding	Moderate, provides good compression for small integers	Moderate, as bit-wise decoding is required
PForDelta Compression	High, particularly for sparse postings lists	Moderate to High, involves block-wise decompression
Variable Byte Encoding	High, especially for large corpora with frequent terms	Low, as decoding is efficient

Table 4: Comparison of Compression Techniques for Inverted Indexes

minimizes the memory overhead per node while preserving concurrency in lookups.

Another model is replicate-and-partition, where each node has a full or partial copy of the entire index, but queries are still split among nodes based on concurrency scheduling. Here, each query can be processed in parallel across multiple nodes, with results merged at the end. The trade-off involves memory duplication versus lower cross-node communication overhead. For massive knowledge bases, partial replication might suffice, balancing redundancy with disk and memory constraints [56] [57] [58] [59].

Parallel processing also benefits from pipeline strategies. A query can be parsed, its relevant index segments identified, and each segment can be processed in parallel. Logical operations, such as intersection of posting lists or application of inference rules, can likewise be parallelized. For example, if a query references multiple symbols $x_1, \ldots, x_m$, one might want the intersection:

$$\text{InvIndex}(x_1) \cap \text{InvIndex}(x_2) \cap \cdots \cap \text{InvIndex}(x_m).$$

Distributing the intersection operation across multiple processors accelerates response times significantly, especially for large intersection sets.

A comparison of various parallel query execution strategies is shown in the table below, considering their trade-offs in memory efficiency, communication overhead, and processing speed.

Another key optimization in parallel processing is load balancing. Workloads should be evenly distributed across processing nodes to prevent bottlenecks. Dynamic load balancing strategies continuously monitor query execution times and reallocate resources accordingly. Adaptive partitioning, in which frequently accessed symbols or relations are replicated across multiple nodes, enhances responsiveness by ensuring popular queries are handled with minimal latency [60] [61] [62].

Beyond traditional parallel processing, modern frameworks leverage distributed computing platforms such as Apache Spark, TensorFlow, and Ray to manage large-scale query execution. Spark-based architectures, for instance, implement resilient distributed datasets (RDDs) that allow efficient parallel execution with fault tolerance. When executing a large-scale query set, Spark distributes tasks as independent partitions, executing them concurrently while preserving data consistency through lineage tracking.

For graph-based knowledge retrieval, parallel breadth-first search (BFS) algorithms provide efficient traversal mechanisms. Given a query that involves multi-hop reasoning, parallel BFS can explore multiple paths simultaneously, significantly reducing retrieval latency. Consider a knowledge graph query that seeks all facts reachable within n hops from a given node x. A naive sequential approach requires processing each hop in a linear fashion, but a parallel BFS distributes node expansions across multiple workers:

$$\text{Reachable}(x, n) = \bigcup_{i=1}^{n} \text{Expand}(\text{Reachable}(x, i-1)).$$

This technique ensures that the search space is explored in parallel, dramatically improving efficiency for large-scale commonsense reasoning tasks.

To further improve efficiency, indexing techniques such as locality-sensitive hashing (LSH) can be incorporated into parallel query execution. LSH approximates nearest-neighbor searches in high-dimensional spaces, allowing similar queries to be grouped and executed in batch mode. This is particularly beneficial for queries that involve semantic similarity computations, such as retrieving facts related to a given concept based on vector embeddings.

The effectiveness of parallel processing frameworks is highly dependent on the underlying hardware architecture. GPU-accelerated query execution, for instance, leverages thousands of cores to process multiple index lookups in parallel. This is especially useful for deep learning-based knowledge

Parallel Execution Strategy	Memory Efficiency	Processing Speed
Sharded Indexing	High, as each node stores only a subset of data	Moderate, depends on balanced distribution of workload
Replicated Indexing	Low, due to full index duplication across nodes	High, minimal communication required for query execution
Distributed Query Processing	Moderate, with adaptive partitioning of workload	High, as queries are executed concurrently across nodes
Pipeline Parallelism	High, as tasks are divided into logical stages	High, suitable for deep query processing workflows

Table 5: Comparison of Parallel Query Execution Strategies

representations, where embeddings are stored in high-dimensional tensor structures. A GPU-based retrieval system can efficiently compute similarity scores between query vectors and indexed facts, significantly improving response times [63] [64].
The table below summarizes different hardware acceleration techniques for parallel query execution, comparing their computational throughput and scalability.
Modern parallel frameworks often utilize GPU acceleration or streaming multiprocessors to handle large batches of queries simultaneously.
GPU-accelerated libraries for set intersection or dictionary lookups can yield order-of-magnitude improvements in throughput. However, these gains come with programming complexities related to memory transfers, caching, and thread synchronization. Performance engineering requires careful balancing of concurrency, data movement overhead, and indexing data structure design [65] [66] [67].

4 | Data Structures and Algorithmic Optimization

Data structures employed for large-scale commonsense indexing must strike a balance between space efficiency, fast lookups, and update flexibility. While inverted files remain a canonical approach, their implementations can vary significantly in both logical organization and low-level representation. A typical inverted file includes a lexicon mapping each symbol to its posting list, where each posting entry references a knowledge base fact or a smaller index block.
Let $L(x)$ denote the posting list for symbol x. Suppose $|L(x)| = n_x$. When n_x is large, a naive traversal of the entire list for each query can be costly. Hence, advanced data structures use skip lists or tree-based indexing over the posting lists to enable logarithmic-time partial lookups. Specifically, a skip-list approach might store pointers at regular intervals, allowing the search process to jump quickly to relevant positions without scanning each element. Mathematically, if $L(x)$ is sorted (e.g., by fact ID or by hash of the tuple (s, r, o)), then a query involving x can use a binary search approach. If we wish to combine results from multiple symbols $x_1, x_2, \ldots, x_m$, we compute:

$$\bigcap_{i=1}^{m} L(x_i).$$

For intersection, specialized algorithms such as the Galloping (or exponential) search exploit sorted lists to reduce complexity. Parallelizing this intersection in a multi-threaded environment can yield near-linear speedups when carefully implemented.
Memory management is equally critical. If we let $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ be all facts in the knowledge base, each fact might appear in multiple posting lists. High redundancy can inflate memory usage. Therefore, fact references can be compressed via integer encoding, block-level compression, or dictionary-based compression. For example, if the average length of a posting list is large, dictionary-based approaches can store the differences between consecutive fact IDs rather than storing each fact ID in full. Symbolic representation might further be compressed if synonyms or identical structures appear repeatedly [68] [69] [70].
Beyond the inverted file, alternative index structures like tries, Patricia trees, or graph-based indices can be relevant if the knowledge base exhibits certain hierarchical or lexical patterns. For symbolic data, a trie might capture the decomposition of symbols into subunits, potentially improving prefix-based queries. Yet, for purely semantic queries such as $\text{IsA}(x, \texttt{Animal})$, a trie of symbol strings might offer marginal benefit compared to a well-structured inverted file.

Hardware Acceleration Method	Computational Throughput	Scalability
Multi-Core CPU Processing	Moderate, constrained by core count	High, scales well across cloud-based architectures
GPU-Accelerated Execution	High, due to thousands of parallel cores	Moderate, limited by memory bandwidth constraints
FPGA-Based Query Execution	Very High, with custom hardware-optimized pipelines	Low, requires specialized hardware and design complexity
Distributed Cloud Computing	High, with elastic scaling of resources	Very High, dynamically allocates resources based on workload

Table 6: Comparison of Hardware Acceleration Techniques for Parallel Query Execution

Algorithmic optimizations also target query scheduling. For a multi-query scenario, we can group queries by overlapping symbol usage, enabling batched or shared index scans. Similarly, caching frequent queries or partial intersection results can be beneficial if the knowledge base experiences repeated patterns of user requests. If a system frequently processes queries for the same set of high-level concepts, partial caching of posting list intersections or partial expansions of inference rules can dramatically reduce response times.

5 | Logic-based Reasoning and Structured Representation

Commonsense knowledge repositories frequently incorporate logical rules, constraints, and typed relationships, all of which influence the indexing and search process. Let us define a logical signature $\Theta = \langle \mathcal{C}, \mathcal{R}, \mathcal{F} \rangle$, where $\mathcal{C}$ is a set of concept symbols, $\mathcal{R}$ is a set of relation symbols, and $\mathcal{F}$ is a set of function symbols (if any). A fact in the knowledge base is then an atomic formula $r(c_1, c_2, \ldots, c_k)$ for $r \in \mathcal{R}$ and $c_i \in \mathcal{C}$. Traditional triple-based representations are a special case where $k = 2$. We can formalize constraints using Horn clauses:

$$\forall x_1, \ldots, x_m \, [\, P_1(x_1, \ldots, x_m) \wedge \cdots \wedge P_l(x_1, \ldots, x_m) \rightarrow Q(x_1, \ldots, x_m) \,]$$

with $P_i, Q \in \mathcal{R}$. Such constraints allow inference of new facts from existing ones.

In index construction, partial evaluation of these Horn clauses can be performed to generate expanded indexes. For instance, if the knowledge base includes a statement such as:

$$\forall x, y \, [\, \text{PartOf}(x, y) \rightarrow \text{RelatedTo}(x, y) \,],$$

we might store $\text{RelatedTo}(x, y)$ in the index whenever $\text{PartOf}(x, y)$ is encountered, effectively short-circuiting the inference at query time. However, this leads to index proliferation if there are many or complex rules. An alternative is to store only the raw facts in the index, applying inference rules dynamically upon query execution. This approach pushes the computational load to query time but avoids large-scale data duplication [71] [72] [73].

Moreover, typed relationships can refine the indexing structure. If each concept $c \in \mathcal{C}$ belongs to one or more types in a type hierarchy $\mathcal{H}$, we can index facts at the type level. For example, if Person is a type and every instance x with $\mathit{type}(x, \text{Person})$ also satisfies certain relational constraints, then indexing by type can expedite queries seeking all persons related to a given concept. Symbolically, we might maintain:

$$\text{InvIndex}(\text{Person}) = \{x \mid \mathit{type}(x, \text{Person})\},$$

and for a relation r,

$$\text{InvIndex}(r, \text{Person}) = \{(x, r, y) \mid x \in \text{InvIndex}(\text{Person})\}.$$

This multi-dimensional indexing approach allows queries to incorporate type constraints without scanning all possible facts.

Structured representation in the form of conceptual graphs or ontologies further augments search efficiency. A conceptual graph can be viewed as a bipartite structure linking concept nodes to relation nodes. One can build index entries for each node and edge, enabling advanced graph matching queries. When parallelizing these queries, each subgraph pattern match can be allocated to different processing units, with partial matches merged subsequently. However, large-scale subgraph matching can be extremely expensive, necessitating specialized algorithms like graph partitioning or dynamic programming. Even so, an efficient inverted index remains a valuable starting point for limiting the search space to candidate nodes and edges relevant to the query pattern [74] [75] [76].

6 | Performance Evaluation and Scalability Considerations

Assessing the efficiency of parallel inverted indexing and retrieval for commonsense knowledge demands a combination of theoretical and empirical methods. Theoretical bounds can be derived by analyzing the complexities of indexing, intersection, and inference operations. Let $N = |\mathcal{K}|$ denote the total number of facts, and let $|\Sigma|$ be the number of unique symbols. Building the inverted index sequentially requires processing each fact, identifying the constituent symbols, and updating their respective posting lists. The naive construction complexity is $O(N)$ if we assume a constant cost for inserting a new entry into each posting list. In practice, this cost depends on the data structure used (linked list, array, tree) and any compression or skipping mechanisms.

Parallel index construction splits $\mathcal{K}$ into partitions processed by different threads or nodes. If each partition is of size N/p for p parallel units, the naive speedup is close to p, discounting synchronization and merging overhead. Merging partial indexes requires union operations on the sets of symbols, which can be performed in $O(|\Sigma| \log p)$ if done carefully. Alternatively, each node can handle a disjoint subset of symbols, mitigating the need for merges at the expense of more complex query handling.

Query execution time is often dominated by the intersection of posting lists for multiple symbols, especially in logic-based queries that require combining evidence. Suppose a query references m symbols, each with posting list length n_i, $i = 1, \ldots, m$. The worst-case complexity of intersecting all lists is $O(\sum_{i=1}^{m} n_i)$, although advanced algorithms and data structures can reduce this. In a parallel setting with p workers, an ideal scenario might distribute intersection tasks evenly, reducing intersection time to $O(\frac{1}{p} \sum_{i=1}^{m} n_i)$. However, load imbalance can arise if certain symbols have disproportionately large posting lists.

Empirical benchmarks are crucial for evaluating real-world performance. In typical experiments, the knowledge base is scaled from tens of millions to billions of facts. Metrics include average query latency, throughput (queries per second), index size on disk, and in-memory footprint. Additionally, we track the scaling efficiency $E = \frac{T_1}{p \cdot T_p}$, where T_1 is the wall-clock time for a single-threaded run, and T_p is the wall-clock time with p threads or nodes. Perfect linear scalability yields $E = 1$, but in practice, communication overhead, synchronization costs, and data partitioning strategies reduce E.

For large-scale commonsense applications, one must also consider dynamic updates to the knowledge base. New facts may arrive continuously, reflecting new inferences or user contributions (e.g., crowdsourced data). Handling real-time updates in an inverted index can be challenging, as adding or removing facts can cause partial index invalidation. Lock-free or concurrency-friendly data structures (e.g., concurrent hash maps or specialized tries) can mitigate these overheads. In a parallel environment, one can adopt a micro-batch approach, buffering updates and applying them in batches to minimize fragmentation. Alternatively, an approach employing a log-structured merge architecture can keep an in-memory index for recent updates and periodically merge it with the main disk-based index [77] [78] [79].

While parallel processing and robust data structures yield substantial performance gains, it is crucial to balance speed with interpretability and correctness. Overly aggressive compression or partial indexing may omit some valuable inferences. Similarly, asynchronous processing might lead to temporary inconsistencies in distributed settings. Hence, a holistic view—combining concurrency control, fault tolerance, and semantic fidelity—is essential for real-world deployments of commonsense knowledge repositories.

7 | Conclusion

Inverted indexing and parallel processing form a powerful combination for enabling real-time access to large-scale commonsense knowledge repositories. By dissecting symbolic data into tokenized posting lists, search operations can be reduced to index lookups whose efficiency is further amplified by multi-threaded or distributed frameworks. The integration of logic-based reasoning and structured representations brings additional depth to the indexing process, allowing semantically rich data to be stored in a manner that can still be queried with low latency. Throughout this discussion, we have examined how careful data structure selection, compression methods, and algorithmic optimizations can mitigate the overhead associated with extensive posting lists and frequent queries. Parallelization strategies, ranging from distributed memory clusters to GPU acceleration, address the computational intensity of intersecting large lists and evaluating complex inference rules on-the-fly. Our analysis highlights the importance of balancing multiple design trade-offs, including memory redundancy versus inter-node communication, partial versus full replication of indexes, and precomputed

versus dynamic inference expansion.

Experimental observations and theoretical considerations both suggest that these techniques—when orchestrated properly—can deliver near-linear speedups and maintain consistent query performance at scale. Future work could explore the synergy between sophisticated knowledge representation frameworks (e.g., ontologies or conceptual graphs) and state-of-the-art parallel hardware, further pushing the envelope of real-time commonsense retrieval. The integration of structured representations with high-performance computing architectures offers an opportunity to enhance both the efficiency and depth of knowledge inference. By leveraging ontologies, intelligent systems can maintain hierarchical and relational organization of concepts, enabling more precise reasoning over complex knowledge domains. Conceptual graphs, on the other hand, provide an intuitive mechanism for encoding relationships and facilitating inferential reasoning, allowing retrieval engines to traverse and contextualize knowledge in a more human-like manner [80] [81] [82].

The incorporation of neurosymbolic approaches represents another promising avenue. By bridging the gap between symbolic reasoning and sub-symbolic deep learning models, future commonsense retrieval systems could benefit from both interpretability and generalization capabilities. Symbolic methods, such as description logic and rule-based inference, offer transparency and structured query resolution, while neural embeddings and transformer-based models contribute robust pattern recognition and contextual adaptation. The challenge lies in effectively harmonizing these paradigms to enable real-time retrieval that is both semantically rich and computationally efficient. Hybrid architectures that dynamically switch between logic-based reasoning and deep learning-based similarity retrieval could pave the way for more robust and adaptive systems [83] [84] [85].

Furthermore, real-time retrieval necessitates advancements in hardware acceleration strategies. The adoption of domain-specific accelerators, such as tensor processing units (TPUs) and field-programmable gate arrays (FPGAs), can significantly reduce query latency, enabling rapid traversal of large-scale knowledge graphs. Parallelization techniques, including model parallelism and data parallelism, can be leveraged to optimize workloads across distributed environments, ensuring scalable and responsive query execution. Additionally, memory-efficient indexing structures, such as compressed tries and probabilistic filters, can be integrated to minimize storage overhead while preserving retrieval accuracy.

A critical area of future research is the development of self-adaptive retrieval mechanisms capable of learning and refining retrieval strategies based on user interactions. Reinforcement learning-based optimization can be employed to iteratively enhance query ranking models, dynamically adjusting the weighting of retrieved results based on feedback loops. Such mechanisms would enable intelligent systems to evolve and personalize responses over time, aligning retrieval outcomes with human expectations. Moreover, active learning paradigms can be introduced to identify gaps or inconsistencies within knowledge repositories, prompting automated refinement of commonsense databases.

The intersection of commonsense retrieval and multimodal knowledge representation also warrants further investigation. As human cognition is inherently multimodal—integrating textual, visual, auditory, and even sensory information—future retrieval frameworks must transcend unimodal text-based approaches. By incorporating computer vision techniques, speech recognition models, and cross-modal embedding spaces, retrieval systems could achieve a more holistic understanding of commonsense phenomena. For instance, a query about "how to tie a shoelace" could retrieve not only textual explanations but also relevant instructional videos and annotated diagrams, improving user comprehension.

Scalability remains an ever-present concern as commonsense knowledge bases continue to expand. Future retrieval architectures must strike a balance between retrieval efficiency and knowledge breadth. One possible direction is the deployment of hierarchical knowledge caching mechanisms that prioritize frequently accessed or high-utility knowledge fragments while deferring long-tail information retrieval to secondary storage layers. Techniques such as graph partitioning and federated knowledge retrieval could also be explored to optimize distributed query execution while minimizing redundancy across replicated datasets [86] [87] [88].

Additionally, the issue of commonsense reasoning robustness demands further scrutiny. Knowledge bases often contain noisy, incomplete, or even contradictory information, necessitating mechanisms for uncertainty quantification and trustworthiness assessment. Probabilistic logic frameworks, such as Markov logic networks or Bayesian reasoning models, could be integrated to provide confidence scores for retrieved facts. Moreover, adversarial testing methodologies could be developed to systematically evaluate the robustness of commonsense retrieval systems against

ambiguous, misleading, or adversarial queries. Finally, ethical considerations must be prioritized in the advancement of commonsense retrieval methodologies. Bias mitigation strategies should be embedded into retrieval pipelines to ensure that retrieved knowledge reflects diverse perspectives rather than reinforcing societal prejudices. Transparent auditability of retrieval decisions is also crucial, allowing researchers and practitioners to diagnose potential biases or errors in retrieved results. By incorporating fairness-aware retrieval mechanisms and explainable AI techniques, next-generation systems can be designed to uphold ethical AI principles while maintaining retrieval effectiveness [89] [90] [91].

By building upon the foundations set forth here, next-generation intelligent systems can harness massive repositories of commonsense knowledge to achieve more nuanced and responsive decision-making, ultimately advancing the broader objectives of artificial general intelligence. The confluence of sophisticated knowledge representation, high-performance parallel processing, adaptive learning mechanisms, and multimodal reasoning will define the trajectory of future commonsense retrieval research. As AI systems increasingly interact with humans in real-world settings, their ability to efficiently retrieve and reason over commonsense knowledge will play a pivotal role in fostering more natural, context-aware, and trustworthy machine intelligence [92] [93] [94].

References

[1] S. S. Souryal, "Demythelogizing personal loyalty to superiors," *Critical Criminology*, vol. 19, pp. 119–135, May 2011.

[2] D. Borri, D. Camarda, and R. Stufano, "Spatial primitives and knowledge organization in planning and architecture: some experimental notes," *City, Territory and Architecture*, vol. 1, pp. 2–, May 2014.

[3] M. Sidman, "Terrorism as behavior," *Behavior and Social Issues*, vol. 12, pp. 83–89, October 2003.

[4] D. Wilkinson, "Towards an archaeological theory of infrastructure," *Journal of Archaeological Method and Theory*, vol. 26, pp. 1216–1241, December 2018.

[5] R. Dean, "Does neuroscience undermine deontological theory," *Neuroethics*, vol. 3, pp. 43–60, November 2009.

[6] S. Prijic-Samarzija, "Trust and contextualism," *Acta Analytica*, vol. 22, pp. 125–138, July 2007.

[7] F. van Harmelen, A. Herzig, P. Hitzler, and G. Qi, "Preface: Special issue on commonsense reasoning for the semantic web," *Annals of Mathematics and Artificial Intelligence*, vol. 58, pp. 1–2, October 2010.

[8] Y. Zeng, N. Zhong, Y. Wang, Y. Qin, Z. Huang, H. Zhou, Y. Yao, and F. van Harmelen, "User-centric query refinement and processing using granularity-based strategies," *Knowledge and Information Systems*, vol. 27, pp. 419–450, May 2010.

[9] Q. Liu, H. Jiang, Z.-H. Ling, X. Zhu, S. Wei, and Y. Hu, "Combing context and commonsense knowledge through neural networks for solving winograd schema problems," November 2016.

[10] *Toward Generating 3D Games with the Help of Commonsense Knowledge and the Crowd*, September 2014.

[11] A. Sharma and K. Forbus, "Automatic extraction of efficient axiom sets from large knowledge bases," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 27, pp. 1248–1254, 2013.

[12] P. R. Smart, A. Madaan, and W. Hall, "Where the smart things are: social machines and the internet of things," *Phenomenology and the Cognitive Sciences*, vol. 18, pp. 551–575, July 2018.

[13] H. Chen, A. Trouve, K. Murakami, and A. Fukuda, *A Concise Conversion Model for Improving the RDF Expression of ConceptNet Knowledge Base*, pp. 213–221. Germany: Springer International Publishing, November 2017.

[14] H. Qi, "A joint parsing system for visual scene understanding," June 2018.

[15] P. Daly, "An integral approach to health science and healthcare," *Theoretical medicine and bioethics*, vol. 38, pp. 15–40, January 2017.

[16] C. Malaviya, C. Bhagavatula, A. Bosselut, and Y. Choi, "Commonsense knowledge base completion with structural and semantic context," January 2019.

[17] C. Golub, "Expressivism and realist explanations," *Philosophical Studies*, vol. 174, pp. 1385–1409, August 2016.

[18] G. Lakemeyer, "The situation calculus: A case for modal logic," *Journal of Logic, Language and Information*, vol. 19, pp. 431–450, January 2010.

[19] M. Nikravesh, V. Loia, and B. Azvine, "Fuzzy logic and the internet (flint): Internet, world wide web, and search engines," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 6, pp. 287–299, August 2002.

[20] *AAAI Fall Symposium: Commonsense Knowledge - FIRE: Infrastructure for Experience-based Systems with Common Sense*, November 2010.

[21] S. Aditya, "Explainable image understanding using vision and reasoning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, February 2017.

[22] R. Rosati, "Multi-modal nonmonotonic logics of minimal knowledge," *Annals of Mathematics and Artificial Intelligence*, vol. 48, pp. 169–185, June 2007.

[23] G. D. Pinal and B. Waldon, "Modals under epistemic tension," *Natural Language Semantics*, vol. 27, pp. 135–188, March 2019.

[24] *AAMAS - Multiagent environment design in human computation*, May 2011.

[25] S. Jastrzebski, D. Bahdanau, S. Hosseini, M. Noukhovitch, Y. Bengio, and J. C. K. Cheung, "Commonsense mining as knowledge base completion? a study on the impact of novelty," January 2018.

[26] J. Pittard, "Disagreement, reliability, and resilience," *Synthese*, vol. 194, pp. 4389–4409, June 2016.

[27] B. Coersmeier and L. Steindler, "Zeitschriftenschau," *Zeitschrift für allgemeine Wissenschaftstheorie*, vol. 14, pp. 185–212, March 1983.

[28] *Textual inference by combining multiple Logic programming paradigms*, December 2005.

[29] M.-S. Chiu, "Identification and assessment of taiwanese children's conceptions of learning mathematics.," *International Journal of Science and Mathematics Education*, vol. 10, pp. 163–191, February 2011.

[30] A. Sharma and K. Forbus, "Graph traversal methods for reasoning in large knowledge-based systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 27, pp. 1255–1261, 2013.

[31] T. Gowan, "New hobos or neo-romantic fantasy? urban ethnography beyond the neoliberal disconnect," *Qualitative Sociology*, vol. 32, pp. 231–257, June 2009.

[32] F. Kraemer, K. van Overveld, and M. Peterson, "Is there an ethics of algorithms," *Ethics and Information Technology*, vol. 13, pp. 251–260, July 2010.

[33] K. Ma, J. Francis, Q. Lu, E. Nyberg, and A. Oltramari, "Towards generalizable neuro-symbolic systems for commonsense question answering," October 2019.

[34] L. Gannett, "Questions asked and unasked: how by worrying less about the 'really real' philosophers of science might better contribute to debates about genetics and race," *Synthese*, vol. 177, pp. 363–385, November 2010.

[35] V. Kumar, "Moral judgment as a natural kind," *Philosophical Studies*, vol. 172, pp. 2887–2910, February 2015.

[36] J. Yan, C. Wang, W. Cheng, M. Gao, and A. Zhou, "A retrospective of knowledge graphs," *Frontiers of Computer Science*, vol. 12, pp. 55–74, September 2016.

[37] S. Heymans, D. V. Nieuwenborgh, and D. Vermeir, "Conceptual logic programs," *Annals of Mathematics and Artificial Intelligence*, vol. 47, pp. 103–137, September 2006.

[38] A. Sharma and K. Goolsbey, "Identifying useful inference paths in large commonsense knowledge bases by retrograde analysis," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[39] *AAAI Fall Symposium: Commonsense Knowledge - Cross-Domain Scruffy Inference*, November 2010.

[40] D. Karagiannis, *Database and Expert Systems Applications: Proceedings of the International Conference in Berlin, Federal Republic of Germany, 1991.* July 1991.

[41] "Abstracts," *Annals of Behavioral Medicine*, vol. 50, pp. 1–335, March 2016.

[42] M. Barrow, J. McKimm, and S. Gasquoine, "The policy and the practice: early-career doctors and nurses as leaders and followers in the delivery of health care," *Advances in health sciences education : theory and practice*, vol. 16, pp. 17–29, June 2010.

[43] P. B. Thompson and K. P. Whyte, "What happens to environmental philosophy in a wicked world," *Journal of Agricultural and Environmental Ethics*, vol. 25, pp. 485–498, September 2011.

[44] P. Harris, J. Turbill, L. Kervin, and K. Harden-Thew, "Mapping the archive: An examination of research reported in ajll 2000–2005," *The Australian Journal of Language and Literacy*, vol. 33, pp. 173–197, October 2010.

[45] W. Pedrycz, "Evolvable fuzzy systems: some insights and challenges," *Evolving Systems*, vol. 1, pp. 73–82, July 2010.

[46] B. H. Hancock, "Learning how to make life swing," *Qualitative Sociology*, vol. 30, pp. 113–133, April 2007.

[47] A. Sharma and K. M. Goolsbey, "Simulation-based approach to efficient commonsense reasoning in very large knowledge bases," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1360–1367, 2019.

[48] R. J. Varey, T. Wood-Harper, and B. Wood, "A theoretical review of management and information systems using a critical communications theory," *Journal of Information Technology*, vol. 17, pp. 229–239, December 2002.

[49] G. Kern-Isberner, M. Wilhelm, and C. Beierle, "Probabilistic knowledge representation using the principle of maximum entropy and gröbner basis theory," *Annals of Mathematics and Artificial Intelligence*, vol. 79, pp. 163–179, May 2015.

[50] R. M. Sutton, K. M. Douglas, and L. M. McClellan, "Benevolent sexism, perceived health risks, and the inclination to restrict pregnant women's freedoms," *Sex Roles*, vol. 65, pp. 596–605, November 2010.

[51] J. Romero, S. Razniewski, K. Pal, J. Z. Pan, A. Sakhadeo, and G. Weikum, "Commonsense properties from query logs and question answering forums," May 2019.

[52] M. U. Smith, "Current status of research in teaching and learning evolution: Ii. pedagogical issues," *Science & Education*, vol. 19, pp. 539–571, November 2009.

[53] J. P. Delgrande, "On a rule-based interpretation of default conditionals," *Annals of Mathematics and Artificial Intelligence*, vol. 48, pp. 135–167, May 2007.

[54] L. Caronia and A. H. Caron, "Morality in scientific practice: The relevance and risks of situated scientific knowledge in application-oriented social research," *Human Studies*, vol. 42, pp. 451–481, January 2019.

[55] J. Petraglia, "Narrative intervention in behavior and public health," *Journal of health communication*, vol. 12, pp. 493–505, July 2007.

[56] A. G. Baydin, R. L. de Mántaras, and S. Ontañón, "A semantic network-based evolutionary algorithm for modeling memetic evolution and creativity," April 2014.

[57] J. Skott, "Contextualising the notion of 'belief enactment'," *Journal of Mathematics Teacher Education*, vol. 12, pp. 27–46, November 2008.

[58] A. Bosselut and Y. Choi, "Dynamic knowledge graph construction for zero-shot commonsense question answering," November 2019.

[59] D. Bamber, I. R. Goodman, and H. T. Nguyen, "Deduction from conditional knowledge," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 8, pp. 247–255, February 2004.

[60] L. F. Sikos, "Rdf-powered semantic video annotation tools with concept mapping to linked data for next-generation video indexing: a comprehensive review," *Multimedia Tools and Applications*, vol. 76, pp. 14437–14460, August 2016.

[61] M. M. Hedblom, O. Kutz, R. Peñaloza, and G. Guizzardi, "Image schema combinations and complex events," *KI - Künstliche Intelligenz*, vol. 33, pp. 279–291, July 2019.

[62] E. Saad and E. Pontelli, "A new approach to hybrid probabilistic logic programs," *Annals of Mathematics and Artificial Intelligence*, vol. 48, pp. 187–243, May 2007.

[63] D. Arnold, "The deceptive simplicity of nāgārjuna's arguments against motion: Another look at mūlamadhyamakakārikā chapter 2," *Journal of Indian Philosophy*, vol. 40, pp. 553–591, October 2012.

[64] W. Biebuyck, "Food supply and government: A 'victualized' reading of european modernity," *Comparative European Politics*, vol. 14, pp. 477–503, May 2016.

[65] Q. Liu, H. Jiang, Z.-H. Ling, S. Wei, and Y. Hu, "Probabilistic reasoning via deep learning: Neural association models.," March 2016.

[66] J. Xu, R. H. Güting, Y. Zheng, and O. Wolfson, "Moving objects with transportation modes: A survey," *Journal of Computer Science and Technology*, vol. 34, pp. 709–726, July 2019.

[67] R. Harré and J.-P. Llored, "Molecules and mereology," *Foundations of Chemistry*, vol. 15, pp. 127–144, February 2013.

[68] R. Hoekstra and J. Breuker, "Commonsense causal explanation in a legal domain," *Artificial Intelligence and Law*, vol. 15, pp. 281–299, February 2007.

[69] M. Balduccini, M. Gelfond, and M. L. Nogueira, "Answer set based design of knowledge systems," *Annals of Mathematics and Artificial Intelligence*, vol. 47, pp. 183–219, September 2006.

[70] J. Hao, M. Chen, W. Yu, Y. Sun, and W. Wang, "Kdd - universal representation learning of knowledge bases by jointly embedding instances and ontological concepts," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1709–1719, ACM, July 2019.

[71] V. van Themaat and J. R. Wettersten, "Rezensionen," *Zeitschrift für allgemeine Wissenschaftstheorie*, vol. 13, pp. 166–173, March 1982.

[72] N. Weißenberg, R. Gartmann, and A. Voisard, "An ontology-based approach to personalized situation-aware mobile service supply," *GeoInformatica*, vol. 10, pp. 55–90, February 2006.

[73] E. Weydert, "Conditional ranking revision - iterated revision with sets of conditionals," *Journal of Philosophical Logic*, vol. 41, pp. 237–271, July 2011.

[74] M. Finthammer, "Concepts and algorithms for computing maximum entropy distributions for knowledge bases with relational probabilistic conditionals," *KI - Künstliche Intelligenz*, vol. 33, pp. 97–100, December 2018.

[75] P. Z. M. U. Sanchez, "Playing the game: Reform politicians in the cebu traditional political field," *GSTF Journal of Law and Social Sciences*, vol. 5, September 2016.

[76] R. D. Mascio, "Firms' adoption of self-service technology: how managerial beliefs shape co-production decisions," *AMS Review*, vol. 6, pp. 79–97, April 2016.

[77] B. Hou, Z. wei Wang, and R. Ying, "Pesticide residues and wheat farmer's cognition: A china scenario," *Agricultural Research*, vol. 5, pp. 51–63, January 2016.

[78] C. Dawson, "Towards a conceptual profile: Rethinking conceptual mediation in the light of recent cognitive and neuroscientific findings.," *Research in Science Education*, vol. 44, pp. 389–414, November 2013.

[79] I. Varzinczak, "A note on a description logic of concept and role typicality for defeasible reasoning over ontologies," *Logica Universalis*, vol. 12, pp. 297–325, September 2018.

[80] "Book reviews," *Dao*, vol. 6, pp. 301–323, September 2007.

[81] C. R. Mauceri, "Expanding commonsense knowledge bases by learning from image tags," July 2015.

[82] *LREC - Acquiring Lexical Knowledge for Anaphora Resolution.*, May 2002.

[83] *IJCAI - Practical partition-based theorem proving for large knowledge bases*, August 2003.

[84] M. Cataldi, R. Damiano, V. Lombardo, and A. Pizzo, *New Trends of Research in Ontologies and Lexical Resources - Lexical Mediation for Ontology-Based Annotation of Multimedia.* Springer Berlin Heidelberg, September 2012.

[85] R. W. P. Luk, "Understanding scientific study via process modeling," *Foundations of Science*, vol. 15, pp. 49–78, January 2010.

[86] D. Summers-Stay, C. R. Voss, and T. Cassidy, "Using a distributional semantic vector space with a knowledge base for reasoning in uncertain conditions," June 2016.

[87] C. Malaviya, C. Bhagavatula, A. Bosselut, and Y. Choi, "Exploiting structural and semantic context for commonsense knowledge base completion," October 2019.

[88] E. Amir, *Reasoning and decision making*, pp. 191–212. Cambridge University Press, June 2014.

[89] A. Stepanjans and A. Freitas, "Identifying and explaining discriminative attributes," January 2019.

[90] *IJCAI - Resource-bounded crowd-sourcing of commonsense knowledge*, July 2011.

[91] S. M. Dromi and S. D. Stabler, "Good on paper: sociological critique, pragmatism, and secularization theory," *Theory and Society*, vol. 48, pp. 325–350, February 2019.

[92] N. B. Cocchiarella, "Infinity in ontology and mind," *Axiomathes*, vol. 18, pp. 1–24, January 2008.

[93] P. Ein-Dor, *Commonsense Knowledge Representation II*. IGI Global, January 2011.

[94] A. Stepanjans and A. Freitas, "Identifying and explaining discriminative attributes.," September 2019.