big data and cognitive computing

MDPI

Article

Hybrid Method of Organizing Information Search in Logistics Systems Based on Vector-Graph Structure and Large Language Models

Vadim Voloshchuk [1], Yaroslav Melnik [2], Irina Safronenkova [1,*], Egor Lishchenko [2], Oleg Kartashov [3] and Alexander Kozlovskiy [2]

[1] R&D Institute of Robotics and Control Systems, Southern Federal University, 105/42 Bolshaya Sadovaya Str., Rostov-on-Don 344006, Russia; vvoloshchuk@sfedu.ru
[2] Institute of Computer Technologies and Information Security, Southern Federal University, 105/42 Bolshaya Sadovaya Str., Rostov-on-Don 344006, Russia; iamelnik@sfedu.ru (Y.M.); lishchenko@sfedu.ru (E.L.); kozlovskiy@sfedu.ru (A.K.)
[3] The Smart Materials Research Institute, Southern Federal University, 178/24 Sladkova Str., Rostov-on-Don 344090, Russia; okartashov@sfedu.ru
* Correspondence: safronenkova@sfedu.ru

Abstract

In logistics systems, the organization of information retrieval plays a key role in human interaction with technical systems to ensure decision-making speed, route optimization, planning, and resource allocation. At the same time, the efficiency of the logistics system when simultaneously processing large volumes of data and constantly updating it is determined by the speed of processing user requests and the accuracy of the responses provided by the system. Within the retrieval-augmented generation architecture, a hybrid information retrieval method has been proposed, based on the combined use of a vector-graph data representation structure and large language model. Experiments showed that the hybrid method achieved best accuracy rates of 0.24–0.25 (among all considered methods) with enhanced scalability capabilities (when the number of nodes increases fourfold, the time increases only twofold—from 0.09 s to 0.20 s) due to the limitation of the graph traversal area when implementing the graph component of the hybrid search. An optimal range of 30–50 nodes to be traversed was also identified, balancing precision and query processing speed. The findings are of practical value to logistics system developers and supply chain managers aiming to implement high-precision, natural language-based information retrieval in dynamic operational environments.

Keywords: LLM; graph DBMS; vector DBMS; RAG architecture; artificial intelligence; logistics; semantic analysis; precision; latency

Check for updates

Academic Editor: Fabrizio Messina

Received: 5 December 2025
Revised: 9 January 2026
Accepted: 2 February 2026
Published: 5 February 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

In the modern world, artificial intelligence (AI) plays a key role in the development and transformation of various areas of human activity. The implementation of intelligent systems in diverse production processes allows for reducing costs, increasing productivity and labor safety, and improving product quality by effectively automating processes and making decisions [1,2]. One example of the effective implementation of intelligent systems is the logistics sector. The use of intelligent logistics systems allows for the automation of various transportation and warehouse processes, including route planning and optimization, warehouse management, inventory management, fleet management,

demand forecasting, and many others. Another example of the implementation of artificial intelligence technologies in the logistics sector is the use of intelligent robots that enable autonomous delivery of goods by land and air (delivery robots).

It is worth noting that the potential for using AI technologies in the logistics sector has been facilitated by the rapid development and widespread adoption of Internet of Things (IoT) and Industrial Internet of Things (IIoT) technologies [3,4]. The IoT paradigm provides the ability to integrate heterogeneous devices through digital data transmission networks. This allows for the management of various physical objects and logistics processes in real time through human interaction with the logistics system [5–7]. Using natural language as a means of interacting with a technical system seems promising, as it provides convenience and versatility. This can be achieved by integrating logistics systems with large language models (LLMs), which provides intuitive interaction between users and the system and allows for the automatic processing of text datasets (transport documentation, warehouse reports, market trend analysis materials) [8–10]. However, by their nature, LLMs do not have access to the company's operational, closed data and are prone to 'hallucinations.' In the logistics sector, such inaccuracies are particularly hazardous as they can lead to severe real-world consequences. For instance, hallucinated information regarding transport routes or cargo handling procedures can result in direct safety risks. Furthermore, providing incorrect data on delivery timelines or warehouse availability can trigger substantial contractual fines, while errors in processing transport documentation can lead to violations of regulatory compliance and legal complications. Moreover, due to characteristics of logistics systems such as the high dynamism of input data, their significant volume, and heterogeneity, the process of handling large language models requires substantial computing resources, which logistics systems often lack. This can lead to reduced precision and speed in processing user requests, and consequently to a decrease in the overall efficiency of the logistics system.

The traditional Retrieval-Augmented Generation (RAG) architecture [11] partially addresses the problem of hallucinations in autonomous LLMs by providing access to external data sources. However, the issue related to the need to process large amounts of contextual information remains unresolved.

Within the framework of this work, it is proposed to use a vector-graph structure for organizing the external memory of a language model (knowledge base). This approach ensures efficient search through subject-oriented data of the logistics system, to which the LLM does not have access through its internal weights. Unlike traditional RAG, which only retrieves isolated text fragments, the proposed method builds a dynamic context based on the connections of the knowledge graph, allowing the model to provide precision and well-founded answers.

The purpose of this work is to develop a hybrid method for organizing information retrieval, which addresses a key problem of modern logistics systems: the integration of heterogeneous, dynamically changing data and providing intelligent access to it through natural language. This allows for increased precision and speed in processing user requests in logistics systems.

To achieve this purpose, the following research questions were formulated:

- RQ1: How does the integration of a graph structure into a vector-based RAG architecture affect the precision of information retrieval in logistics systems with dynamic data?
- RQ2: What is the impact of the hybrid vector-graph approach on system latency and scalability as the volume of logistics data increases?
- RQ3: What are the optimal hyperparameters (number of nodes to traverse and expansion depth) to balance precision and query processing speed?

Based on the formulated research questions, the following hypotheses were proposed:

- H1: The hybrid vector-graph method will provide significantly higher precision than pure vector or graph-based methods by utilizing structural connections to enrich the semantic context.
- H2: The hybrid method will demonstrate a sublinear growth in latency relative to graph size, provided that the traversal depth and neighbor count are constrained.
- H3: There exists a "saturation point" in the trade-off between precision and speed, where increasing the number of traversed nodes beyond a certain range (30–50) yields diminishing returns in accuracy while significantly increasing latency.

The remainder of this paper is organized as follows. Section 2 reviews the existing solutions for intelligent systems in logistics and the theoretical foundations of RAG. Section 3 presents the proposed hybrid method for organizing information search, detailing the vector-graph data representation. Section 4 describes the experimental setup, visualization of studies, and performance evaluation. Section 5 provides a discussion of the obtained results, while Section 6 concludes the paper by summarizing the findings and outlining directions for future research.

2. Background

2.1. Existing Solutions for the Application of Intelligent Systems in the Field of Logistics

Currently, one of the main directions in the development of logistics is digitalization, including technologies such as the IoT, cloud computing, blockchain, big data and analytics, cyber-physical systems, and AI. It should be noted that AI occupies a special place among the listed technologies, as its application ensures high-quality solutions in the management of various logistics processes. Thus, the use of AI technologies makes it possible to significantly improve the efficiency of warehouses, transportation hubs, and distribution centers, minimizing the number of errors caused by the human factor [12].

Several main areas of AI technology application in the logistics sector are highlighted below, using industrial case studies as illustrative examples of the potential benefits of such systems.

- Delivery route optimization. AI technologies allow the analysis of large volumes of data, including information about road traffic, weather conditions, and other factors affecting delivery. This not only helps reduce the time vehicles spend on the road but also lowers transportation and maintenance costs. As an illustrative instance, reports from DHL indicate that the company managed to reduce delivery times by 15% and cut operating expenses by 10% thanks to the active use of AI technologies [13].
- Automation of warehouse operations. This procedure includes tracking and managing inventory, routing cargo, processing orders and shipments, as well as quality control. The case of Amazon serves as an example, where the implementation of AI-powered robotic systems reportedly allowed for a 20% reduction in operating costs and a fourfold increase in order processing speed [14].
- Demand forecasting and inventory management. AI technologies allow companies to predict enterprise needs and generate well-balanced recommendations for purchasing and moving inventory. This reduces storage costs and lowers the risk of product shortages during peak periods. In another reported case, Walmart demonstrated the potential to reduce excess inventory by 25%, resulting in significant cost savings [15].
- Supply chain management. Machine learning algorithms analyze data on supplies, production capacities, and demand to optimize the entire process from manufacturing to delivery to the end consumer. For instance, reports indicate that Procter & Gamble used AI to manage its global supply chain, which improved its operational performance by 20% [16].

As noted above, a convenient and versatile way of interaction between a logistics system and an operator is communication in natural language. Currently, there are ready-made solutions that implement this function in the form of voice assistants. For example, the Honeywell Vocollect Voice software solution is designed to organize warehouse workflows through voice control [17]. The Manhattan Associates Voice Direct cloud solution is used to automate order processing and optimize warehouse production processes. The platform supports integration with ERP and WMS (Warehouse Management System) systems. The Lucas Systems Voice-Directed Warehouse Operations platform is designed to automatically guide staff actions and analyze their performance. The assistant helps workers effectively allocate resources and minimize equipment downtime [18]. The Blue Yonder Luminate Digital Assistant software application integrates with demand forecasting, supply planning, and transportation route optimization systems. The user interacts with the system by voice, requesting information about order status, cargo location, product stock in the warehouse, and other indicators [19].

However, existing systems do not provide for the processing of large volumes of unstructured, highly dynamic input data, which can be presented in the form of text arrays (transportation documentation, warehouse reports, market trend analysis materials). For example, transportation companies use data from previous trips of a similar type to plan routes; however, the actual road conditions are dynamic and constantly changing. During transportation, various factors can affect the speed of vehicle movement and delivery times: traffic jams and road congestion, changes in weather conditions, accidents. These circumstances inevitably lead to deviations from the planned schedule, delays, or accelerated delivery. Therefore, it is important for transportation companies to promptly monitor changes in road conditions and quickly provide customers with accurate information about the current delivery timelines.

The general principles of interaction between a human and a logistics system are shown in Figure 1.

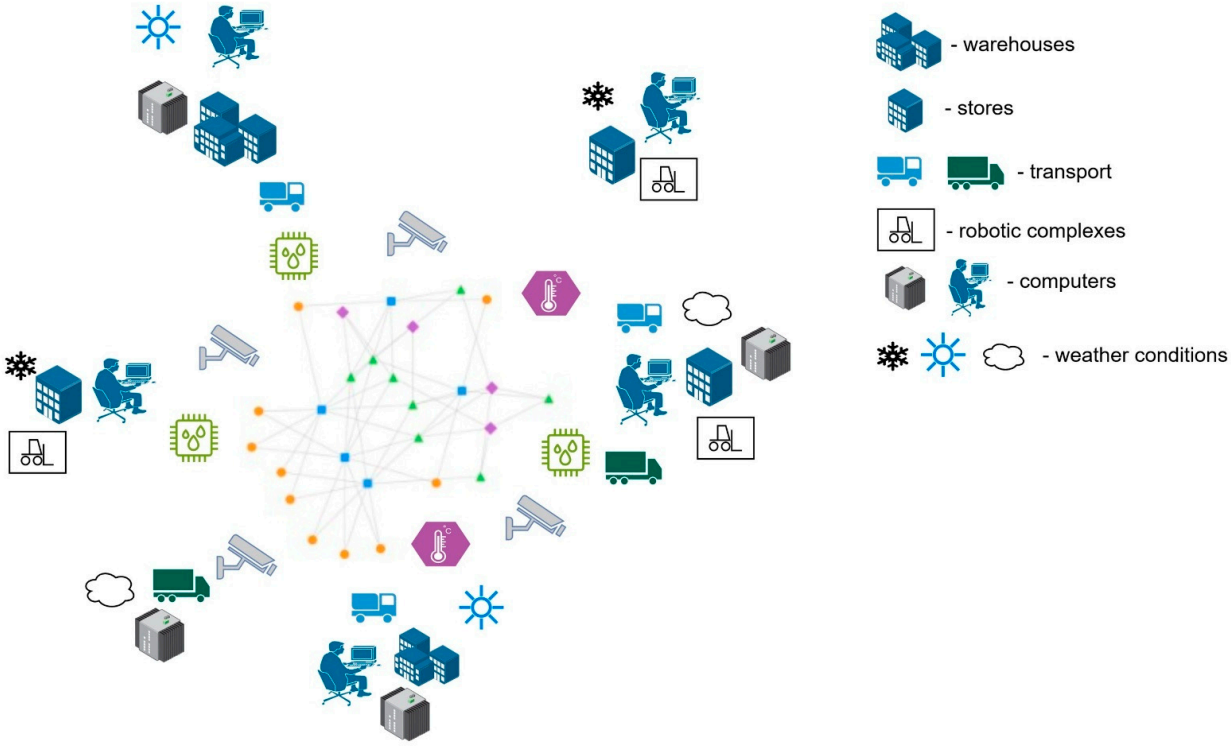


Figure 1. General principles of interaction between a human and a logistics system.

Figure 1 shows the main elements of the environment for interaction between a human and a logistics system. Logistic objects (warehouses, stores, transport, robotic complexes, computers), the connections between them (roads), and weather conditions affecting vehicle movement are depicted as a graph. The system's database stores dynamic data on the traffic situation, which is collected using IIoT and IoT devices. Additionally, historical data is available in data warehouses. Based on the available data, the system generates responses to user queries using intelligent data processing technologies.

It should be noted that in order to ensure interaction between the logistics system and humans in natural language, the system's ability to take context into account in human-system dialogues is an important condition for its effective functioning. Moreover, such systems must provide timely processing of a large volume of unstructured data.

The use of the RAG approach in such systems partially addresses the problem of context generation by enabling the retrieval of data from external sources. However, the issue of timely processing of large volumes of data with limited system resources remains unresolved. Therefore, within the framework of this work, it is proposed to modify the traditional RAG architecture in terms of knowledge retrieval.

2.2. Retrieval-Augmented Generation

RAG is a machine learning technology that combines two key artificial intelligence technologies: information retrieval and text generation. This approach allows generative models to access external data sources, such as knowledge bases, documents, or online resources, which significantly improves the quality of the generated content and reduces hallucinations in autonomous LLMs [20]. Thus, RAG includes 2 components:

- The retriever uses various techniques, including traditional statistical (BM25) and dense vector representations dense passage retrieval (DPR).
- The generator generates contextually relevant responses based on the retrieved knowledge and often uses transformer architecture.

2.2.1. Knowledge Retriever Mechanisms

The mechanisms used by the retriever are an important component of the effective operation of RAG.

The BM25 algorithm [21], developed in the 1970s and 1980s, uses term frequency and inverse document frequency (TF-IDF) to rank documents according to their relevance. There are also known modifications of the BM25 algorithm, including the BM25L algorithm, oriented towards processing long text or variable-length documents [22], BM25F, which differs in its ability to consider the structure of a document and assign different weights to different fields of importance [23], and BM25T allows taking into account not only the logical structure of the document (title, section, paragraph), but also other parameters, for example, bold, italic, center, etc. [24].

These algorithms demonstrate high efficiency in matching keywords; however, they are limited in understanding semantic meaning. Thus, when processing more complex natural language queries that require understanding context, this class of algorithms loses its effectiveness.

Knowledge retriever mechanisms based on DPR were first described in the work [25]. The key idea of this approach is the separate encoding of the query and documents into vector representations. This ensures efficient nearest neighbor search. Unlike algorithms like BM25, DPR takes into account the semantic similarity between the query and documents, which determines its effectiveness in generating answers to open-domain question-answering tasks. After training the retriever on a large corpus of documents and queries,

it is capable of finding documents that are contextually related to the query, even in the absence of identical terms.

In the article [26], a modification called LLM adapted for dense RetrievAl (LLaRA) of the DPR model was proposed, which consists of its integration with pretrained LLMs that were, in turn, adapted for dense vector retrieval. The work [27] presents hybrid generation models with access to parametric (pretrained seq2seq model) and non-parametric memory (dense vector index).

In the work [28], a new method of data augmentation is proposed, aimed at addressing class imbalance in large-scale information retrieval systems. To improve the precision of Dense Passage Retrieval, a token deletion method based on frequency of occurrence in the corpus is used, differing from traditional methods in that it calculates the importance of tokens considering frequency at both the passage and corpus levels, leading to more efficient token removal.

Currently, several industrial frameworks provide established baselines for graph-based retrieval. Microsoft Research's GraphRAG focuses on global community detection and hierarchical summarization of large text corpora. Similarly, LangChain and LlamaIndex (via KnowledgeGraphIndex) offer tools for structured triple-based extraction and basic graph-vector lookups. While effective for general-purpose knowledge management, these frameworks often face challenges in real-time logistics environments where the data is highly dynamic and requires localized, low-latency reasoning over specific infrastructure connections.

2.2.2. Data Generation Mechanisms

The data generation module in RAG plays a key role in generating final output, by integrating the retrieved information with the input query. The generator synthesizes information obtained from external sources into contextually coherent responses. At the core of the generator lies LLM.

For example, the BERT model (Bidirectional Encoder Representations from Transformers) is a technique in the field of natural language processing (NLP) that uses transformer encoder blocks to predict missing words in a given text. BERT differs from earlier models in that it is able to account for context from both the left and right, considering both previous and subsequent words to generate vector representations for individual words and entire sentences. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create models for a wide range of tasks, such as question answering and language inference, without significant modifications to the architecture [29].

Next, the BERT model has been developed in numerous studies. For example, in the work [30], a new model called Contextualized Late Interaction Retrieval (ColBERT) was introduced, which is an adaptation of the BERT language model. Unlike the traditional early interaction approach, the ColBERT model applies deep learning after a preliminary evaluation of the relevance between the query and the document. This allows for reduced computational costs and improved result quality.

The work [31] also presents an improved version of the BERT model—Robustly Optimized BERT Pre-training Approach (RoBERTa). Unlike the traditional BERT approach, which applied fixed random masking of individual words at the beginning of each training epoch, RoBERTa uses a new technical solution that provides dynamic token masking during pre-training. This significantly improved the model's performance in the task of understanding and extracting meaning from complex and large volumes of text.

A Lite BERT (ALBERT) was also developed based on BERT with the aim of reducing computational overhead and memory usage while maintaining performance. This is

achieved through the implementation of two key parameter reduction techniques: factorized embedding parameterization and parameter sharing across layers [32].

The BigBird model allows processing long sequences (up to 4096), significantly surpassing BERT, which can handle no more than 512 tokens. The BigBird model uses a sparse attention mechanism that reduces the quadratic dependency on the length of the sequence to a linear one, enabling efficient processing of long documents [33].

Bidirectional and Auto-Regressive Transformer (BART) is a data generation model in the field of NLP. This model is based on a combination of bidirectional encoding architecture (such as BERT) and autoregressive decoding architecture (such as GPT). Thus, BART ensures the consistency of the generated text by taking into account the context from both past and future tokens. The BART model uses a pre-training and fine-tuning approach for specific natural language processing tasks [34].

T5 (Text-to-Text Transfer Transformer) is a model based on the idea of transfer learning. This model treats each natural language processing task as a "text-to-text" transformation task. T5 uses an encoder–decoder architecture similar to sequence-to-sequence models based on transformers, converting all natural language processing tasks into a text-to-text format. This structure allows the T5 model to be adapted for a wide range of tasks. In the context of RAG, T5 models show better results compared to traditional models: GPT-3 and BART [35].

However, despite all the existing advantages of RAG and active research in this field, this approach has a number of limitations.

- Quality and relevance of search. Improving search precision remains a serious problem and a topic for further research.
- Coherence. RAG models often have difficulties integrating extracted knowledge into a coherent, contextually relevant text.
- Scalability. As retriever mechanisms are directly related to processing large amounts of external databases, this requires the use of efficient search algorithms. High computational costs are also one of the constraints for deploying RAG in real-time systems with limited resources.

2.3. Data Representation Structures

The choice of the way data is presented in automated systems plays an important role because it determines the possible search methods applied to them. In modern logistics systems, various data representation structures and processing methods are used, each of which has its advantages and limitations. In particular, an important task is the optimization of logistics routes for delivery robots and manipulator robots.

2.3.1. Graph Representation of Data

The use of graph structures for data representation allows for high efficiency when working with connected data, including the logistics sector. Such structures provide an accurate accounting of logistical constraints, including the road network, weather conditions, and traffic congestion. Successful examples of implementing graph data structures include the databases Neo4j and Amazon Neptune [36,37]. However, graph structures are characterized by the lack of semantic analysis capabilities for text information, which is critically important when processing transportation invoices, warehouse reports, and other unstructured data.

2.3.2. Vector Representation of Data

An alternative to graph representation is vector representation. Vector representations of data are successfully used in databases like Pinecone, Weaviate, and FAISS. They allow

for fast searching based on semantic similarity; however, they do not take into account the structural relationships between objects in the logistics network. As a result, such systems demonstrate low precision in routing, as they are unable to correctly interpret logistics dependencies [38,39].

Combining vector search with LLMs, as in the RAG architecture, partially addresses the issue of semantic analysis but leads to significant computational costs due to the need for the language model to process large volumes of context. Moreover, such systems remain poorly adapted to dynamically changing conditions.

3. Hybrid Method of Organizing Information Search

In this paper, we propose a hybrid method for organizing information search to improve precision when processing user requests in natural language in logistics systems. This method consists of the combined use of a hybrid vector-graph data representation structure [40–42] and LLMs. Vector-graphic representation of data allows for efficient search in large arrays of dynamically changing data in terms of the precision of the results obtained. Furthermore, this method does not require large computational resources for its implementation compared to traditional search methods.

The proposed solution builds on the existing GraphRAG concept, which is actively being researched by major companies. Its distinguishing feature is its application in the area of dynamically updated logistics network structures. This method of information retrieval also addresses the problem of traditional logistics systems, which rely on rigid interfaces or standard keyword searches. They lose efficiency when processing unstructured data and complex, multi-step dependencies. The proposed use of natural language in conjunction with a hybrid vector-graph structure allows us to overcome this limitation by combining the understanding of text semantics (for example, cargo descriptions in invoices) with consideration of the strict topology of the logistics network. Unlike traditional RAG models, which ignore the structure of connections, or pure graph databases, which are incapable of semantic analysis, this approach provides high precision and contextually justified responses without a critical increase in computational costs. This simplifies access to analytics, allowing operators without technical skills to instantly uncover hidden dependencies, such as the impact of weather conditions on specific deliveries, through an intuitive dialogue. This makes it possible to create a solution that significantly enhances the adaptability of supply chain management, reducing response time to incidents and minimizing risks caused by the 'information blindness' of standard algorithms.

To further clarify the novelty of the proposed approach, it is necessary to distinguish it from traditional GraphRAG methodologies and their industrial implementations, such as Microsoft Research's GraphRAG, LangChain, and LlamaIndex. While standard implementations, particularly the hierarchical community-based approach of Microsoft GraphRAG, typically rely on global community detection and summarization of the entire graph, the proposed hybrid method focuses on localized seed-based expansion.

The key differences lie in three main aspects:

1. Computational Efficiency: Unlike traditional GraphRAG (exemplified by Microsoft Research's GraphRAG), which often requires intensive re-indexing when graph data changes, the proposed method is optimized for high dynamism through light-weight subgraph retrieval. This makes it more suitable for real-time logistics operations where processing new events must occur without the latency of a full global index rebuild.
2. Context Construction: While general-purpose graph-retrieval frameworks (such as LangChain or LlamaIndex) often rely on simple semantic similarity or direct triple lookups, a tunable scoring logic using alpha, beta, gamma, and delta parameters is

introduced. This allows the system to balance semantic similarity with topological connectivity and node importance, which is critical for logistics hubs.

3. Multi-component Ranking: Tunable scoring logic is introduced using alpha, beta, and gamma parameters, allowing the system to balance semantic similarity with topological connectivity.

This localized and parameterized nature addresses a specific requirement of logistics that general frameworks often overlook: the need for real-time reasoning in complex, fast-changing networks where both high speed and structural connectivity are paramount.

Figure 2 shows the general scheme of the hybrid method implementation. The construction of a graph model allows you to formally describe the subject area: identify its main concepts and the connections between them. Further transformation of the graph model of the domain and the user query into a vector-graph structure is necessary to reduce the number of calculations associated with the subsequent processing of such structures.

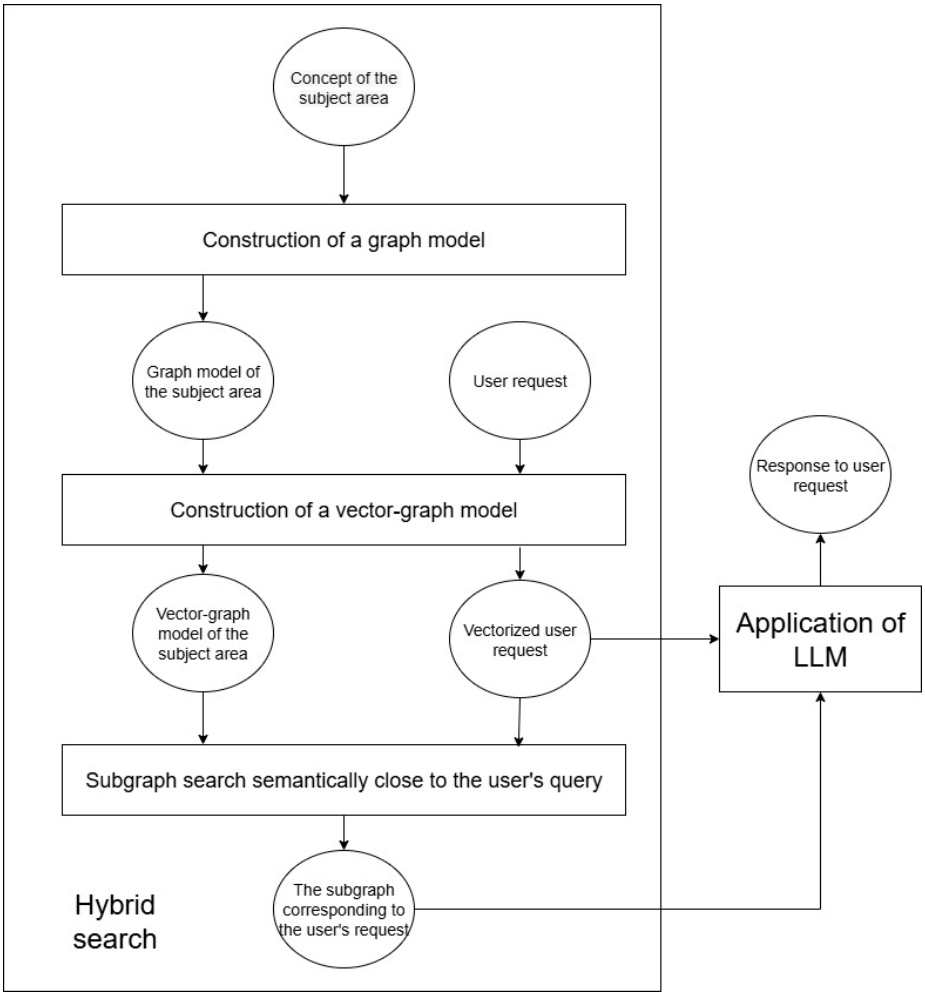


Figure 2. The scheme of the hybrid method.

Thus, the search area is reduced for further processing of graph-vector structures with a large language model, which allows increasing the precision of results under limited computational resources.

3.1. Vector-Graph Representation of the State of a Logistics System

For the effective organization of search in complex and dynamic systems, such as logistics, it is necessary to have a data representation capable of overcoming the fundamental limitations of both purely graph-based and purely vector-based approaches. The proposed

vector-graph structure combines two complementary approaches: topological modeling for preserving structural connections and semantic indexing for analyzing textual information.

The graph component is responsible for the accurate and visual representation of the logistics network topology. In this structure, the graph vertices correspond to the key entities of the subject area, which can be divided into static objects (for example, stores, warehouses, road segments) and dynamic factors (external events such as weather conditions, traffic jams, or accidents). The graph edges, in turn, model explicit, deterministic relationships between these entities: for example, the "serves" relation between a warehouse and a store, or "affects" between a weather phenomenon and a road segment. The main function of the graph component is to ensure structural integrity and precise modeling of explicit dependencies, which is critically important for routing tasks and supply chain analysis.

The vector component solves the problem of semantic analysis, which is not accessible to traditional graph databases. Each graph node containing a textual description (for example, shop_desc, weather_desc) undergoes a vectorization process—transforming into a multidimensional numerical vector (embedding) using a pre-trained language model such as SBERT. This allows determining semantic similarity between nodes as a distance in this multidimensional vector space. A user query, formulated in natural language, is also converted into a vector in the same coordinate system, allowing relevant nodes to be found based on the meaning of the query, not just keyword matching.

Figure 3 shows a three-dimensional visualization (obtained, for example, using the principal component method) of this multidimensional space. Each vertex, such as warehouse_1, has a dual nature: it has both explicit structural connections with other nodes (edges) and a unique position in semantic space (coordinates). The user's query vector is also visualized in this system, clearly demonstrating its semantic proximity to various clusters of nodes.

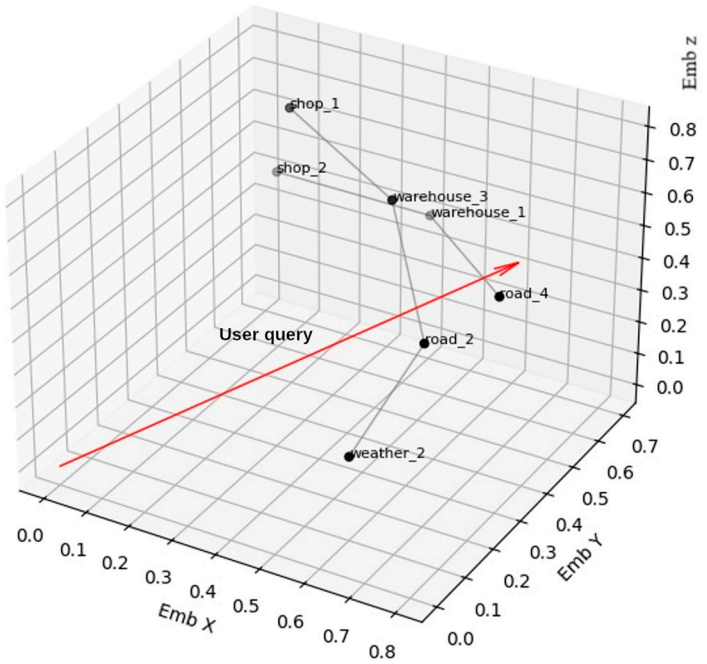


Figure 3. Example of a vector-graph structure describing the state of a logistics system taking into account the user query.

The red arrow represents the vector of the user query, which has been transformed into a multidimensional embedding space. The direction and position of this arrow demonstrate the semantic proximity of the query to specific clusters of logistics objects, such as

warehouses or road segments, illustrating how the system identifies the most relevant nodes based on their coordinates in the semantic space.

The key advantage of such a hybrid structure lies in the synergy of these two approaches. It allows for the formulation and efficient processing of complex queries that combine semantic and topological conditions, for example: "Find a warehouse semantically similar to a 'high-speed distribution hub' that is structurally connected to 'store No. 3'." It is precisely this hybrid nature of the data that requires the development of an appropriate search method capable of operating with both semantics and graph topology simultaneously to provide the most accurate and relevant results.

3.2. Use of the Hybrid Method in the Logistics Sector

The application of a hybrid method combining a vector-graph structure and large language models in the context of logistics is considered. As shown in the diagram (Figure 4), the nodes of the graph contain information about warehouses, stores, and other points of purchase/sale, as well as auxiliary information about roads, weather, and other factors affecting delivery. The edges represent logical connections, such as purchase requests, routes, and the impact of external events. Information search queries are generated depending on the presence or absence of certain data at the nodes, for example, when it is necessary to build an optimal route or adjust it in an unforeseen situation.

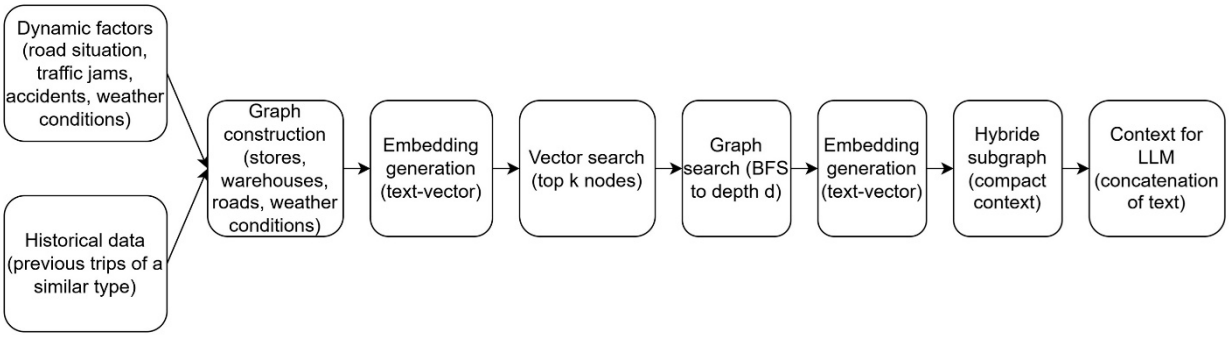


Figure 4. Implementation scheme of the vector-graph approach.

Each node is vectorized and stored as an embedding, which allows for fast retrieval of a relevant subgraph that serves as context for the LLM and improves the precision of the results obtained.

The hybrid search architecture implemented in this study combines semantics from SBERT embeddings and graph topology in a multi-stage process. In the first stage, the query text is scanned for direct mentions of identifiers, such as 'store N' or 'warehouse N'. If such a pattern is found, the system immediately returns a BFS subgraph around this node to a specified depth, which is an efficient solution for precise queries.

If direct references are absent, the vector stage of 'seed' selection begins. The query is converted into an embedding, and using cosine similarity, the seed_k closest nodes in the graph are found. These nodes serve as starting points for the graph expansion stage. At this stage, a breadth-first search (BFS) is launched from each 'seed' to a depth of depth, and all discovered nodes are combined into a single local subgraph. This allows semantically relevant nodes to be enriched with entities structurally connected to them.

Next is the final ranking stage, where nodes in the semi-complete subgraph are re-ranked based on a combined score. This score is a weighted sum of the semantic similarity to the query (alpha parameter) and the structural closeness to the "seeds" (beta parameter). To ensure a comprehensive and balanced ranking, the scoring logic incorporates four weighted components. While alpha (α) and beta (β) account for semantic similarity and structural proximity to the search seeds, the gamma (γ) parameter introduces the local

importance of a node within the retrieved subgraph (calculated via Personalized PageRank). This allows the system to identify nodes that are most 'influential' in the specific context of the query. Finally, the delta (δ) parameter accounts for the global importance of a node based on its overall centrality in the entire logistics network (normalized degree). The conceptual motivation for using γ and δ is to prevent the system from prioritizing isolated nodes that may be semantically similar but lack operational significance in the broader logistics chain. From this ranked list, the top_k most relevant nodes are selected to form the "rough" result. For more complex tasks, an optional beam stage may be applied, where the result is expanded to max_total nodes using a priority queue. Regardless of whether the beam stage is used, the final step performs the ultimate ranking of all retrieved nodes to generate the final output.

The behavior of the method is flexibly configurable using key hyperparameters such as seed_k, depth, top_k, alpha, beta, and max_total. This multi-step approach allows the language model to process only a small but highly relevant subgraph, significantly reducing the computational load compared to approaches based on pure RAG.

4. Experimental Studies

4.1. Visualization of Experimental Studies

The scheme of the subsequent experiment is illustrated using a small visual example. Figures 5–7 show the results of implementing three search methods (graph-based, vector-based, hybrid) on a logistic system graph with a small number of entities for clarity.

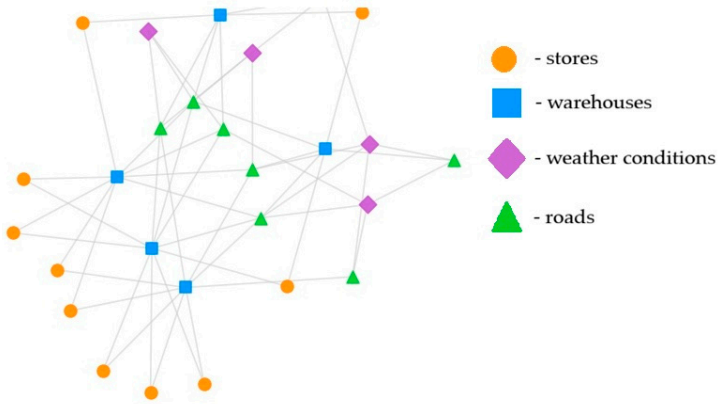
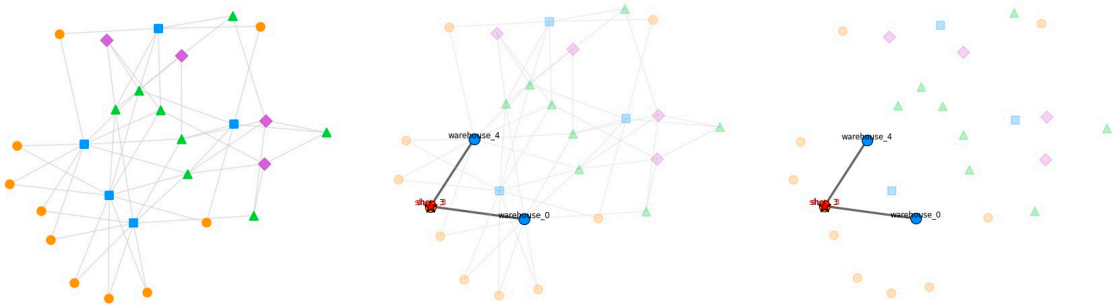


Figure 5. The original graph structure of the logistics system.



(**a**) (**b**) (**c**)

Figure 6. Visualization of the graph search method: (**a**) The initial graph of the logistics system; (**b**) The found subgraph according to the query; (**c**) The final connected subgraph extracted using the breadth-first search method.

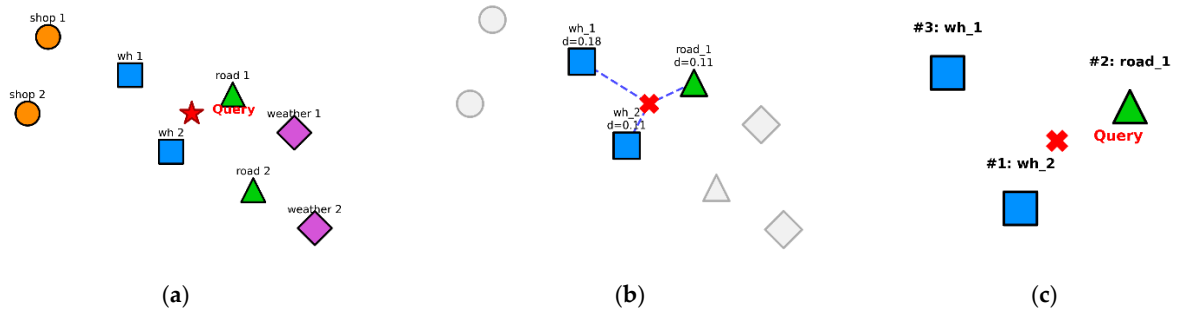


Figure 7. Visualization of the vector search method: (**a**) Vectorized vertices of the source graph and a vectorized user query; (**b**) Visualization of the vector search method; (**c**) Final selection of the Top-K nodes with the highest cosine similarity (on an enlarged scale).

Generated synthetic data based on a predefined schema, including entities such as stores, warehouses, roads, and weather conditions, as well as their relationships, represented in the form of supply requests, delivery routes, and the impact of weather factors. Data generation was carried out using specified parameters for the number of entities (10 stores, 5 warehouses, 8 roads, 4 weather conditions) and a fixed set of relationships between them, providing a structured basis for subsequent analysis. The resulting data were converted into a graph structure using the graph construction module (Figure 5), where each entity was assigned a textual description (e.g., a description of a store or the weather). The logical schema of the logistics graph and the parameters of the visual test case are summarized in Table 1.

Table 1. Logical schema of the logistics graph and parameters of the visual test case.

Entity Type (Node)	Quantity (Visual)	Associated Relationships (Edges)	Linked Entities (Target/Source)
Store (Shop)	10	supply_request, delivery_path	Warehouse, Road Segment
Warehouse	5	supply_request, delivery_route	Store, Road Segment
Road Segment	8	delivery_route, delivery_path, weather_effect	Warehouse, Store, Weather
Weather Condition	4	weather_effect	Road Segment
Total	27 nodes		

To ensure the objectivity of the evaluation, the query generation process was automated based on the graph's known topology. For each query, the 'ground truth' relevant nodes were determined by direct relational extraction: the system identified all entities that strictly satisfy the structural logic of the request (e.g., all warehouses linked to a specific store). This approach provides a rigorous baseline, as the 'correct' answers are mathematically derived from the database schema rather than being manually assigned.

At the same time, queries are being generated that include tasks to determine the warehouses serving specific stores, as well as the assessment of road delays depending on weather conditions (rain, snow, calm, wind) for all available weather identifiers, with predefined relevant answers for each category of queries. The following types of queries to the system are possible: "Find an alternative supplier for store No. 5, which stores 'fragile cargo' and is located within 20 km, avoiding routes with 'roadworks'", "Find all transport invoices from the 10th to the 15th of last month where the 'Cargo' field contains a description similar to 'fragile electronics', and the driver's notes mention 'shaking' or 'bad road'", "On which delivery days last week were delays justified in reports with wording close to 'road blocked due to weather conditions', and which routes were affected?", "Which warehouses are currently overcrowded with goods described in market reports as

'declining demand' or 'outdated collection'?", "To which stores selling 'dairy products' will deliveries be delayed today due to an accident on the M-4 highway?" and so on.

The evaluation utilized 12 distinct query categories, stratified by complexity to ensure a balanced representation of logistics scenarios. The distribution of these queries, ranging from single-hop lookups to complex multi-hop reasoning, is detailed in Table 2.

Table 2. Distribution and complexity of the test query set.

Query Category	Complexity Level	Path Depth	Distribution (%)
Direct Entity Lookup	Single-hop	1	20%
Route/Supplier Assessment	Multi-hop	2	30%
Weather & Delay Impact	Multi-hop	3	40%
Complex Multi-factor Search	Multi-hop	3+	10%

The graph method was implemented through beam search algorithm, a heuristic search algorithm that explores a graph by expanding promising nodes in a limited set. Beam search uses breadth-first search to build its search tree [43]. Some of the results of the ray search are shown in Figure 6.

Vector search. Converting the query text into an embedding vector of fixed dimension. All nodes of the graph are also vectorized. The vectorized vertices of the source graph and the vectorized user query are shown in Figure 7.

Then the cosine similarities of the query vector with all embeddings of graph nodes are calculated. This allows you to select the set of top K nodes with the greatest similarity and form the "core" of the relevant subgraph. The minimum value among these K similarities will be the threshold for adding to the subgraph only those nodes (neighbors) whose semantic similarity to the query is not lower than this threshold.

The hybrid approach combined both methods, integrating a vector search with a top-k parameter (the number of vertices relevant to the query) and a graph search with a depth parameter, which allowed for both semantic proximity and structural relationships (Figure 8).

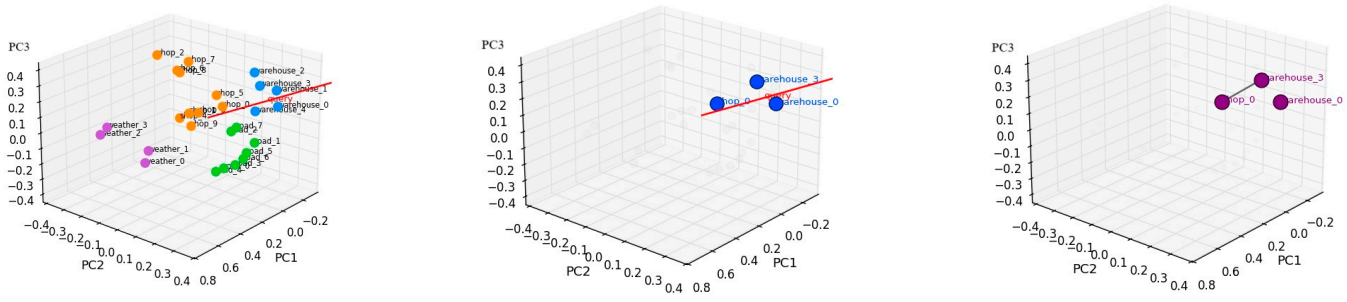


Figure 8. Visualization of the vector-graph method: (**a**) Vector-graph representation of the logistics system; (**b**) The found subgraph according to the query; (**c**) Final Top-K node selection after structural expansion and combined ranking.

The algorithm implementing hybrid search, combining semantics from SBERT embeddings and graph topology, is presented in Algorithm 1:

Algorithm 1. Hybrid search with coarse limiting [HybridSearch(query, G, index, params)]

```
INPUT: query (text), G (graph), index (vector index),
       params (seed_k, depth, alpha, beta, gamma, delta, max_neighbors)
OUTPUT: rankedNodes (list of nodes)
directId = MatchPattern(query, "{shop|warehouse}_{id}");
if directId is not null and G.Contains(directId)
    return BFSSearch(G, directId, params.depth);
queryEmb = Vectorize(query);
seeds = index.FindNearest(queryEmb, params.seed_k);
subgraphNodes = Copy(seeds);
frontier = Copy(seeds);
for d = 1 to params.depth
    nextFrontier = NewSet();
    for node in frontier
        neighbors = G.GetNeighbors(node);
        if Length(neighbors) > params.max_neighbors
            neighbors = Slice(neighbors, 0, params.max_neighbors);
        nextFrontier.AddAll(neighbors);
    subgraphNodes.AddAll(nextFrontier);
    frontier = nextFrontier;
vectorCandidates = index.FindNearest(queryEmb, 20);
allCandidates = Union(subgraphNodes, vectorCandidates);
pprScores = ComputePersonalizedPageRank(G.Subgraph(subgraphNodes), seeds);
rankedList = NewList();
for node in allCandidates
    sim = CosineSimilarity(queryEmb, node.embedding);
    dist = GetShortestPath(subgraphNodes, seeds, node);
    prox = (dist < Infinity) ? (1.0/(1.0 + dist)) : 0.0;
    ppr = pprScores[node]; //Local importance
    glob = node.normalizedDegree; //Global importance
    score = (params.alpha * sim) +
            (params.beta * prox) +
            (params.gamma * ppr)  +
            (params.delta * glob);
    rankedList.Add(node, score);Sort(rankedList, Descending);
return SelectTop(rankedList, params.top_k);
```

To determine the set of possible values for each hyperparameter, we will use the Grid search method. We define the input data for conducting the experiment: top_k ∈ {5, 25, 50, 100}, depth ∈ {1, 2}, max_total ∈ {20, 50}, $\alpha \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, $\beta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$, eval_k = 3—the number of nodes we take for the evaluation. The heat map (Figure 9) reflects the results of the grid search for the hybrid search method.

Figure 9 shows that the hybrid method achieves the highest precision when α is in the range [0.75; 0.95] and β is in [0.05; 0.15]. The dominance of α means that in the queries under consideration, semantic similarity already ensures that relevant nodes appear in the top 3, while the topological component (β) acts as a soft structural regularizer and a mechanism for local context expansion. In practice, this means that the graph component is effective with limited local traversal (restriction on the number of neighbors), providing connectivity and context completeness at low computational cost, whereas a significant increase in P@3 should not be expected.

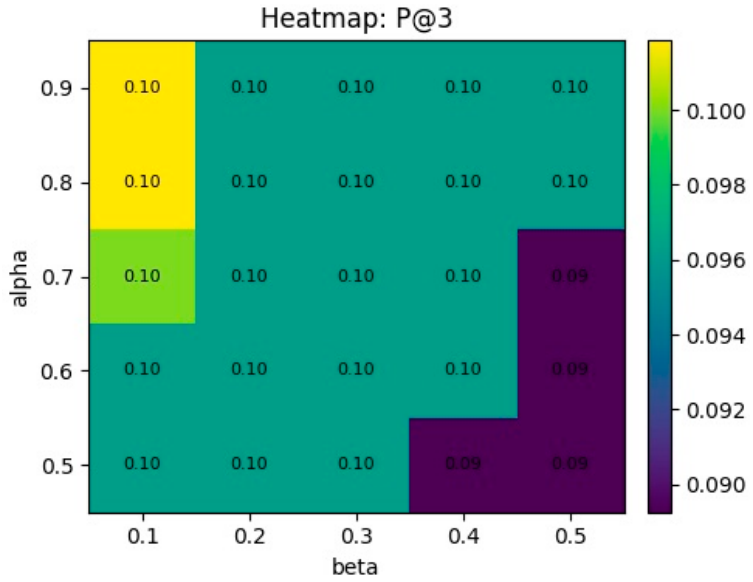


Figure 9. Heatmap of precision (for top k = 3).

4.2. Experiment on a Large-Scale Graph and Performance Evaluation

To evaluate the performance and precision of the developed hybrid method, a series of experiments was conducted on synthetic datasets simulating logistics structures of various scales. The experimental setup included three configurations (small scale, medium scale, large scale) represented in Table 3. A comparison was made between three approaches: vector search, graph search, and the hybrid vector-graph method. The main metrics were precision (Precision@k) and latency.

Table 3. Quantitative characteristics of the datasets used for performance evaluation.

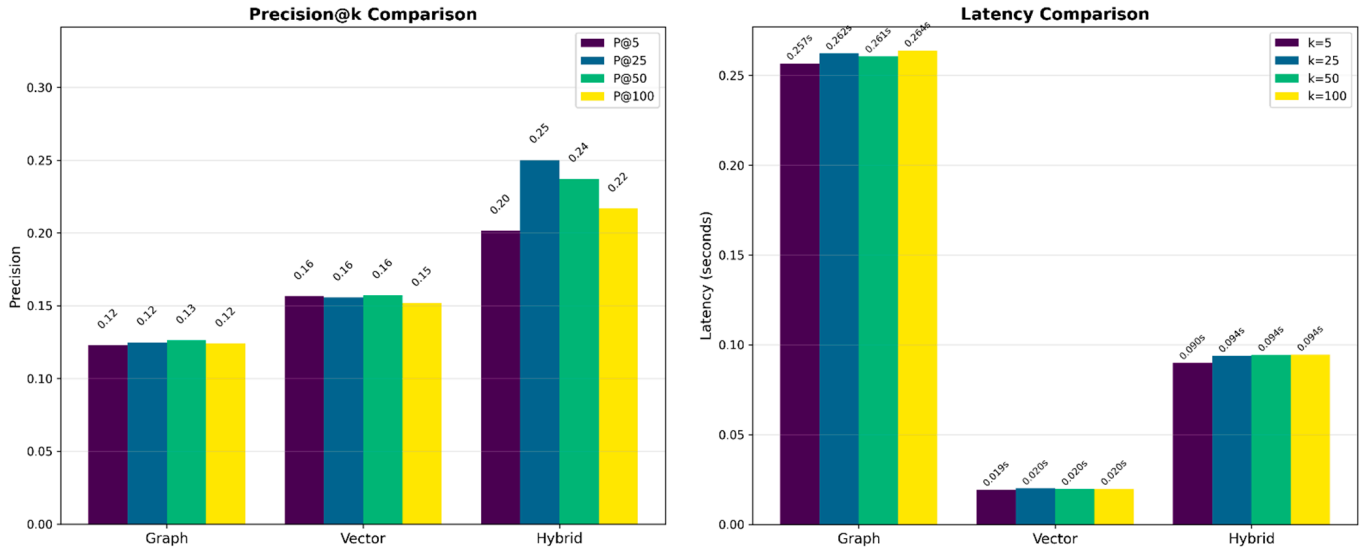
Parameter	Small Scale	Medium Scale	Large Scale
Number of Stores (Shops)	250	500	1000
Number of Warehouses	125	250	500
Number of Road Segments	150	300	600
Number of Weather Types	5	10	20
Total Nodes	530	1060	2120
Total Edges	43,630	175,010	700,020

To ensure the reproducibility and statistical stability of the results, the synthetic data generation process was governed by a multi-seed experimental design. Each scale configuration was tested over multiple iterations, where each run utilized a unique random seed (calculated as a base seed plus the iteration index) to introduce controlled unpredictability into the graph topology and node attributes.

Connectivity between entities was established using specific distribution patterns: for example, the number of supply requests per store followed a uniform distribution within the range of 1–3, simulating realistic logistics constraints. Textual descriptions for each node were sampled from a structured vocabulary with added stochastic noise to mimic the variability of natural language documentation. The reported precision and latency metrics represent the mean values across these iterations. The low variance observed between runs with different seeds confirms the stability of the proposed hybrid method and ensures that the performance gains are not dependent on a specific graph configuration.

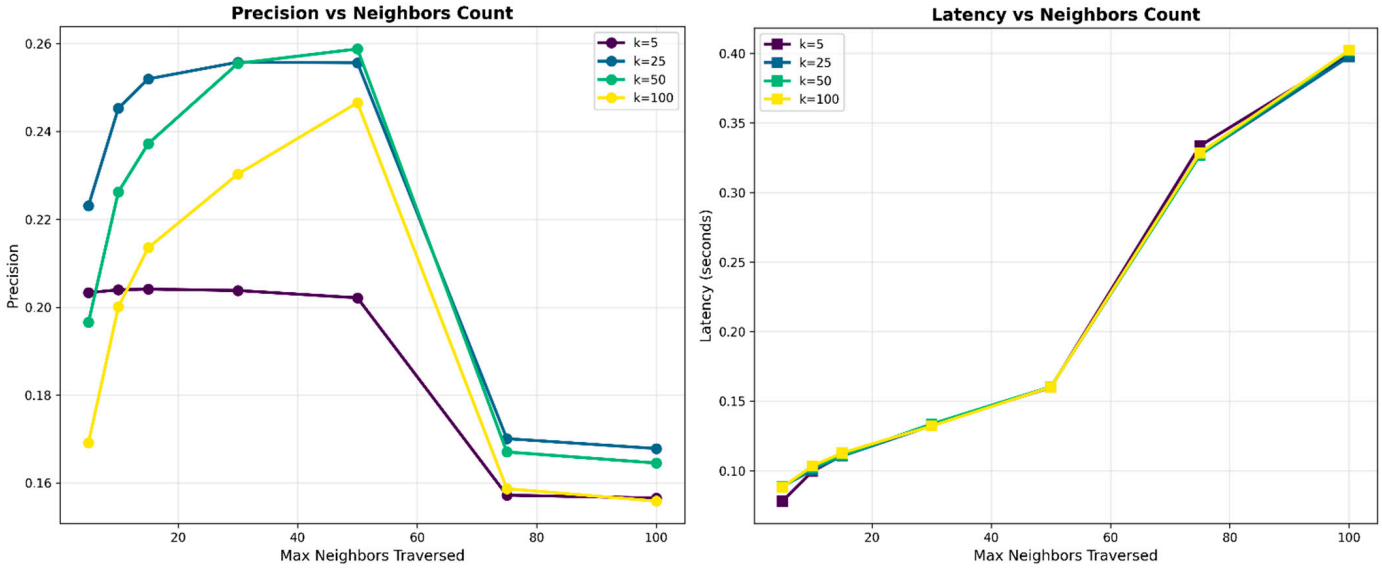
The key distinction of the experimental phase was the generation of more complex, multi-step queries (multi-hop queries) that require searching through chains of several related entities. Examples of such queries include: “What weather affects delivery to store

15?" To answer such a query, the system must sequentially identify the store, then the warehouse associated with it, determine the roads used for delivery, and finally identify the weather conditions affecting those roads (i.e., the sequence Store → Warehouse → Road → Weather). Additionally, this experiment introduced latency assessment—the time spent processing a single query—as a critically important performance metric. The results of the experiments are shown in Figures 10–15 and Tables 4–9.



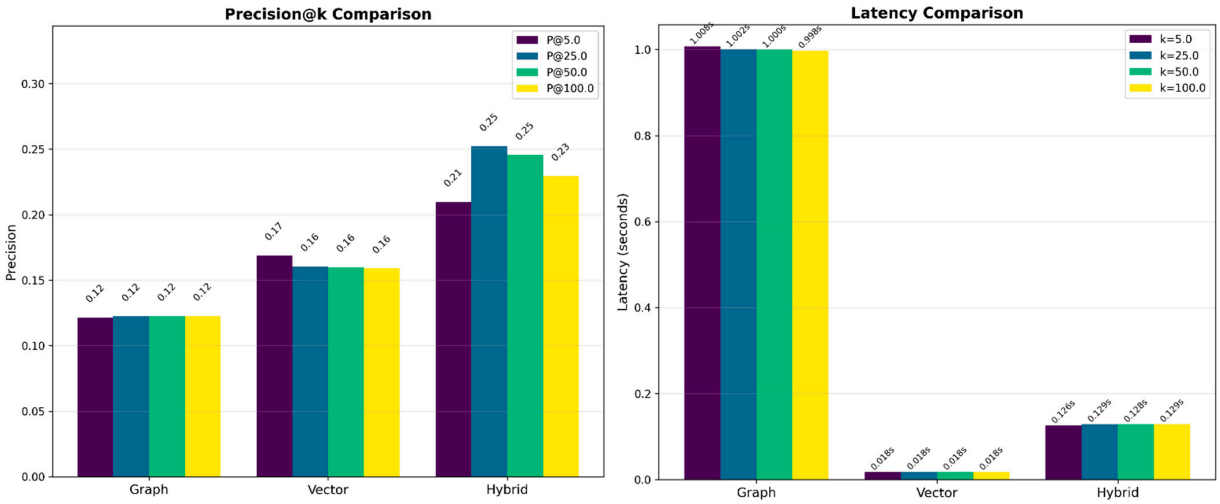
(**a**) (**b**)

Figure 10. Comparative chart of precision (**a**) and latency (**b**) for three methods on a small-scale graph (530 nodes, 43,630 edges).



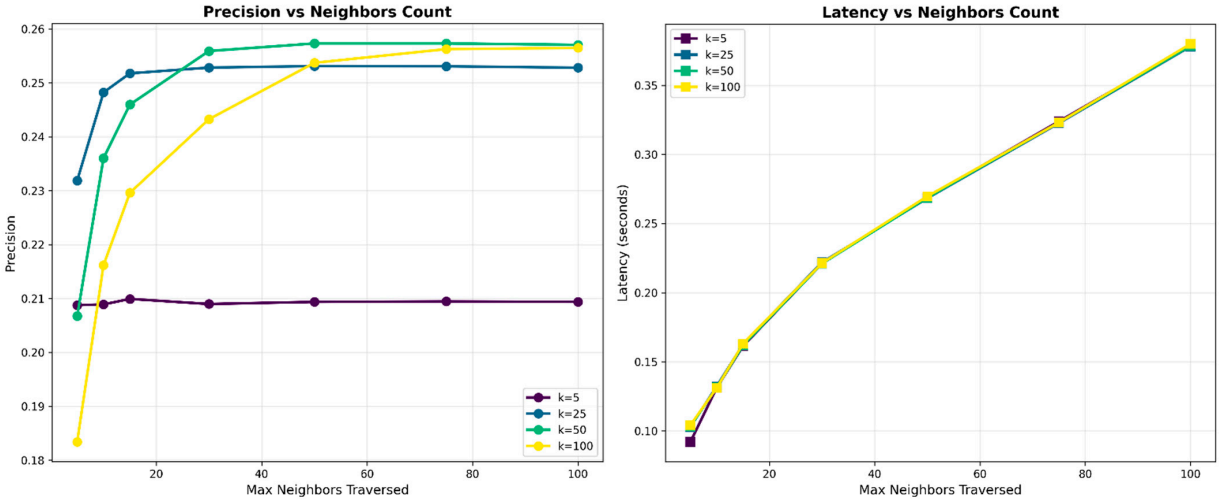
(**a**) (**b**)

Figure 11. Graphs of the dependencies between the number of neighbors viewed when traversing the graph, search precision (**a**), and time delay (**b**) (530 nodes, 43,630 edges).



(**a**) (**b**)

Figure 12. Comparative chart of precision (**a**) and latency (**b**) for three methods on a medium-scale graph (1060 nodes, 175,010 edges).

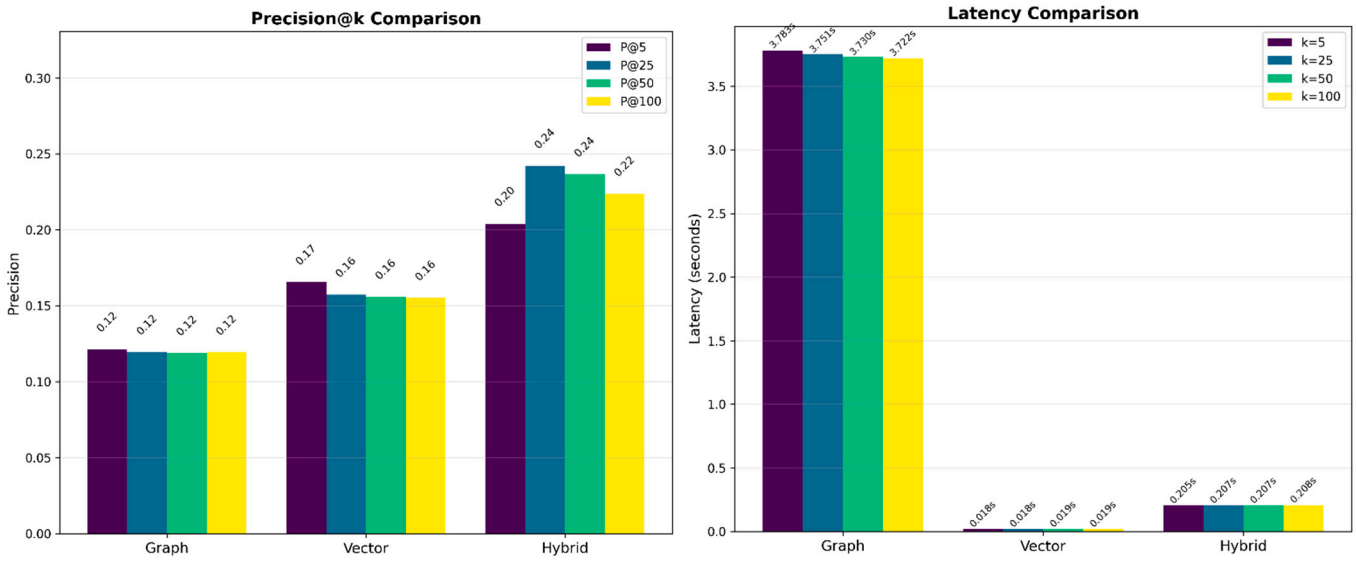


(**a**) (**b**)

Figure 13. Graphs of the dependencies between the number of neighbors viewed during graph traversal, search precision (**a**), and time delay (**b**) (1060 nodes, 175,010 edges).

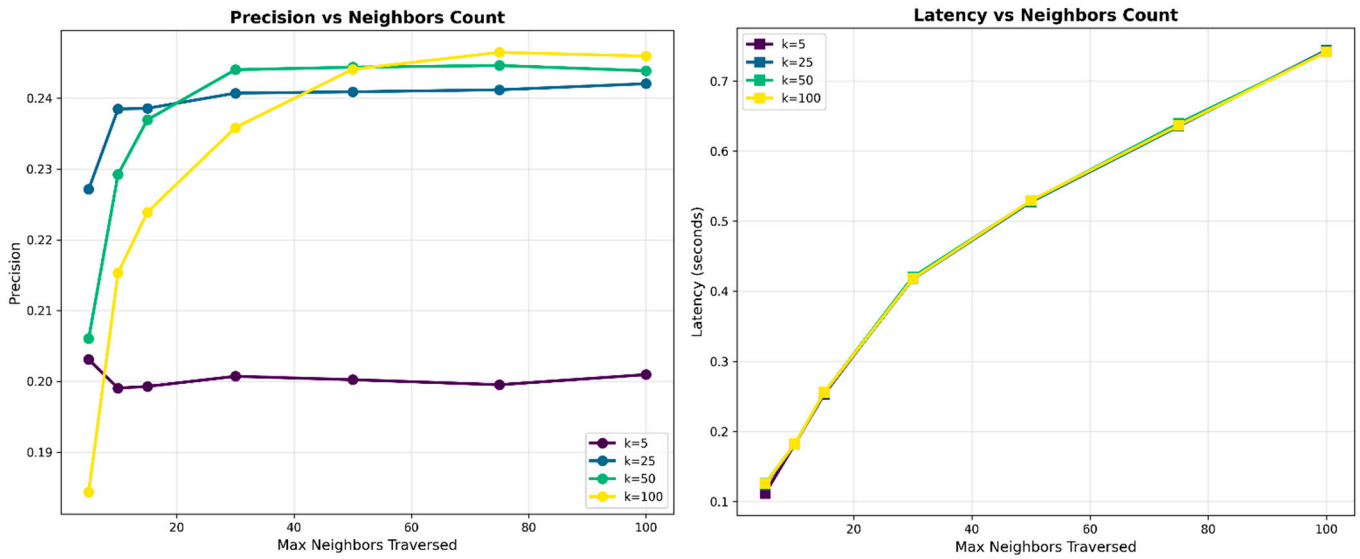
Table 4. Performance evaluation and statistical stability of graph-based search methodologies on a small-scale graph (530 nodes, 43,630 edges).

Method	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
graph	5	0.123	0.056	0.2566	0.0156
graph	25	0.125	0.050	0.2622	0.0163
graph	50	0.126	0.048	0.2606	0.0158
graph	100	0.124	0.047	0.2638	0.0163
vector	5	0.156	0.049	0.0192	0.0001
vector	25	0.156	0.032	0.0199	0.0001
vector	50	0.157	0.027	0.0196	0.0001
vector	100	0.152	0.020	0.0198	0.0001
hybrid	5	0.201	0.055	0.0898	0.0002
hybrid	25	0.250	0.068	0.0938	0.0002
hybrid	50	0.237	0.061	0.0942	0.0002
hybrid	100	0.217	0.049	0.0945	0.0002



(**a**) (**b**)

Figure 14. Comparative chart of precision (**a**) and latency (**b**) for three methods on a large-scale graph (2120 nodes, 700,020 edges).



(**a**) (**b**)

Figure 15. Graphs of the dependencies between the number of neighbors viewed during graph traversal, search precision (**a**), and time delay (**b**) (2120 nodes, 700,020 edges).

Table 5. Statistical summary of the dependencies between the number of neighbors viewed when traversing the graph, search precision, and time delay (530 nodes, 43,630 edges).

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
5	5	0.203	0.059	0.0781	0.0002
10	5	0.204	0.061	0.0995	0.0001
15	5	0.204	0.061	0.1111	0.0002
30	5	0.204	0.061	0.1328	0.0005
50	5	0.202	0.061	0.1599	0.0008
75	5	0.157	0.043	0.3336	0.0051
100	5	0.157	0.043	0.3996	0.0066

Table 5. *Cont.*

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
5	25	0.223	0.055	0.0887	0.0001
10	25	0.245	0.065	0.1006	0.0002
15	25	0.252	0.068	0.1106	0.0002
30	25	0.256	0.069	0.1325	0.0004
50	25	0.256	0.069	0.1602	0.0008
75	25	0.170	0.038	0.3272	0.0052
100	25	0.168	0.038	0.3978	0.0068
5	50	0.197	0.041	0.0884	0.0001
10	50	0.226	0.057	0.1022	0.0004
15	50	0.237	0.061	0.1111	0.0003
30	50	0.255	0.066	0.1336	0.0005
50	50	0.259	0.067	0.1604	0.0008
75	50	0.167	0.034	0.3271	0.0052
100	50	0.165	0.033	0.4016	0.0070
5	100	0.169	0.029	0.0883	0.0001
10	100	0.200	0.041	0.1035	0.0002
15	100	0.214	0.048	0.1128	0.0003
30	100	0.230	0.050	0.1322	0.0004
50	100	0.247	0.053	0.1602	0.0007
75	100	0.159	0.026	0.3285	0.0054
100	100	0.156	0.025	0.4023	0.0069

Table 6. Performance evaluation and statistical stability of graph-based search methodologies on a medium-scale graph (1060 nodes, 175,010 edges).

Method	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
graph	5	0.122	0.057	1.0076	0.2367
graph	25	0.122	0.048	1.0017	0.2329
graph	50	0.122	0.047	1.0001	0.2321
graph	100	0.123	0.047	0.9981	0.2313
vector	5	0.169	0.053	0.0176	0.0001
vector	25	0.161	0.035	0.0180	0.0001
vector	50	0.160	0.031	0.0179	0.0001
vector	100	0.159	0.027	0.0178	0.0001
hybrid	5	0.210	0.062	0.1258	0.0007
hybrid	25	0.252	0.071	0.1287	0.0008
hybrid	50	0.246	0.068	0.1285	0.0008
hybrid	100	0.229	0.061	0.1289	0.0008

Table 7. Statistical summary of the dependencies between the number of neighbors viewed when traversing the graph, search precision, and time delay (1060 nodes, 175,010 edges).

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
5	5	0.209	0.061	0.0919	0.0002
10	5	0.209	0.061	0.1311	0.0004
15	5	0.210	0.061	0.1614	0.0008
30	5	0.209	0.061	0.2211	0.0026
50	5	0.209	0.061	0.2691	0.0035
75	5	0.209	0.061	0.3240	0.0043
100	5	0.209	0.061	0.3788	0.0051
5	25	0.232	0.060	0.1028	0.0002
10	25	0.248	0.069	0.1322	0.0006
15	25	0.252	0.071	0.1627	0.0011
30	25	0.253	0.071	0.2219	0.0028
50	25	0.253	0.071	0.2683	0.0034
75	25	0.253	0.071	0.3227	0.0043

Table 7. *Cont.*

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
100	25	0.253	0.071	0.3782	0.0051
5	50	0.207	0.047	0.1032	0.0002
10	50	0.236	0.063	0.1313	0.0005
15	50	0.246	0.068	0.1619	0.0009
30	50	0.256	0.072	0.2210	0.0026
50	50	0.257	0.072	0.2687	0.0035
75	50	0.257	0.072	0.3232	0.0043
100	50	0.257	0.072	0.3786	0.0051
5	100	0.183	0.038	0.1041	0.0002
10	100	0.216	0.053	0.1311	0.0004
15	100	0.230	0.061	0.1631	0.0010
30	100	0.243	0.065	0.2214	0.0026
50	100	0.254	0.069	0.2697	0.0044
75	100	0.256	0.069	0.3231	0.0043
100	100	0.256	0.069	0.3800	0.0053

Table 8. Performance evaluation and statistical stability of graph-based search methodologies on a large-scale graph (2120 nodes, 700,020 edges).

Method	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
graph	5	0.121	0.058	3.7832	3.3198
graph	25	0.119	0.048	3.7509	3.2691
graph	50	0.119	0.046	3.7299	3.2154
graph	100	0.119	0.046	3.7220	3.2014
vector	5	0.166	0.052	0.0180	0.0001
vector	25	0.157	0.035	0.0185	0.0001
vector	50	0.156	0.032	0.0185	0.0001
vector	100	0.155	0.029	0.0187	0.0006
hybrid	5	0.204	0.060	0.2046	0.0031
hybrid	25	0.242	0.068	0.2075	0.0029
hybrid	50	0.237	0.065	0.2073	0.0037
hybrid	100	0.224	0.061	0.2083	0.0069

Table 9. Statistical summary of the dependencies between the number of neighbors viewed when traversing the graph, search precision, and time delay (2120 nodes, 700,020 edges).

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
5	5	0.203	0.065	0.1115	0.0003
10	5	0.199	0.060	0.1817	0.0011
15	5	0.199	0.061	0.2531	0.0038
30	5	0.201	0.061	0.4195	0.0172
50	5	0.200	0.061	0.5280	0.0249
75	5	0.200	0.061	0.6365	0.0282
100	5	0.201	0.061	0.7437	0.0291
5	25	0.227	0.061	0.1239	0.0003
10	25	0.238	0.066	0.1817	0.0016
15	25	0.239	0.066	0.2546	0.0038
30	25	0.241	0.067	0.4174	0.0168
50	25	0.241	0.067	0.5276	0.0242
75	25	0.241	0.068	0.6353	0.0275
100	25	0.242	0.068	0.7439	0.0277
5	50	0.206	0.051	0.1257	0.0003
10	50	0.229	0.062	0.1819	0.0011
15	50	0.237	0.065	0.2558	0.0038

Table 9. *Cont.*

Max_ Neighbors	k	Precision_Mean	Precision_Disp	Latency_Mean	Latency_Disp
30	50	0.244	0.069	0.4203	0.0170
50	50	0.244	0.068	0.5288	0.0254
75	50	0.245	0.069	0.6395	0.0279
100	50	0.244	0.068	0.7416	0.0286
5	100	0.184	0.043	0.1252	0.0003
10	100	0.215	0.056	0.1822	0.0011
15	100	0.224	0.060	0.2559	0.0037
30	100	0.236	0.065	0.4180	0.0166
50	100	0.244	0.068	0.5294	0.0242
75	100	0.246	0.069	0.6369	0.0275
100	100	0.246	0.069	0.7416	0.0279

The data in Figures 10–15 and Tables 4–9 demonstrate the effectiveness of the proposed approach: with only slightly longer processing time compared to vector search, it provides significantly higher accuracy. A detailed discussion of the experimental results is provided in Section 5. Discussion.

5. Discussion

Analysis of search precision results showed that the precision metrics of the methods remain unchanged when varying the data volume. In all three scenarios, the graph method demonstrated a precision score of 0.12. Vector search showed results in the range of 0.16–0.17. The hybrid method achieved precision scores of 0.24–0.25 in all conducted tests. Differences in the results are due to the mechanisms of the algorithms: the graph method is limited by topological connectivity, the vector method relies on semantic similarity, and the hybrid method combines both approaches, allowing for the extraction of context-dependent data.

The selection of Precision@k as the primary metric is justified by the specific nature of logistics queries, where the 'ground truth' typically consists of a very limited set of nodes (1–5 entities). In such scenarios, Precision@k and Recall@k are highly correlated, as successfully retrieving the correct structural path inherently ensures high coverage. From an operational RAG perspective, precision is the critical factor, as minimizing 'noise' within the LLM's context window directly prevents hallucinations and reduces the computational overhead of processing irrelevant data.

It should be noted that the absolute precision values of 0.24–0.25 may appear low compared to standard classification tasks. However, in the context of multi-hop information retrieval within large-scale logistics graphs (over 700,000 edges), these results represent a significant improvement over traditional methods. In practical logistics decision-making, the retriever's primary role in RAG architecture is to provide a sufficient context for the Large Language Model. A precision of 0.25 indicates that the relevant information is successfully captured within the top-k results, effectively reducing the search space by several orders of magnitude. Given the high noise-to-signal ratio in dynamic logistics networks, such precision is sufficient for an LLM to synthesize an accurate final answer, significantly reducing the cognitive load on human operators who would otherwise have to manually trace these connections. The deliberate stratification of the test set, with 80% of queries being multi-hop, creates a challenging environment that prevents any single method from having an unfair advantage. While simple vector search performs well on single-hop queries, the hybrid approach's ability to maintain precision across higher path depths (2 and 3+) confirms its specific utility for complex logistics reasoning where semantic similarity alone is insufficient. This precision gain is critical for the 'last-mile' task of answering

generation, as the hybrid method provides the LLM with a connected subgraph rather than isolated text embeddings. By delivering explicit structural relations (edges) between logistics entities, the system enables more robust 'chain-of-thought' reasoning during the generation phase. This structured context acts as a grounding mechanism, ensuring that the retrieved facts are presented to the model within their operational hierarchy. Consequently, the improvement in retrieval quality directly supports the synthesis of more accurate, context-aware responses and significantly reduces the risk of logical hallucinations.

The weighting of the topological component (β) in the heatmap, while ranging only between 0.05 and 0.15, has a measurable impact on the final results. Data indicates that shifting from a pure vector search (β = 0) to the hybrid configuration improves precision from approximately 0.16 to 0.25. This suggests that the 10–15% weight attributed to the graph component accounts for a nearly 56% relative increase in retrieval performance. In multi-hop logistics scenarios, this weight ensures that entities with a direct structural link to the search 'seeds' are prioritized over nodes that are semantically similar but structurally irrelevant. Thus, the computational cost of the graph traversal is a necessary trade-off for the resulting precision gain, as it allows the system to filter semantic candidates through the actual constraints of the logistics network.

When analyzing temporal characteristics, a dependency of delay on data volume was identified. Vector search showed a stable response time of about 0.02 s regardless of the sample size. Graph search demonstrated an increase in delay as the graph size grew: from 0.26 s in the small scenario, to 1.00 s in the medium, and 3.78 s in the large scenario. The time growth dynamics for the graph method are linear or superlinear. The hybrid method showed lower delay growth rates: 0.09 s, 0.13 s, and 0.20 s for small, medium, and large scales, respectively. With a fourfold increase in the number of nodes (from 250 to 1000 stores), the hybrid method's response time increased approximately 2.2 times, indicating a sublinear dependence of computational complexity on data volume.

A study of the influence of hyperparameters, in particular the maximum number of neighbors considered, showed the presence of a saturation point in the precision metric. Increasing the number of neighbors to a range of 30–50 leads to a growth in the Precision metric, after which the value stabilizes and reaches a plateau. At the same time, the query execution time grows linearly with respect to the number of nodes examined. Based on the obtained data, a range for the number of neighboring nodes to be traversed in the implementation of the hybrid method has been determined, ensuring a precision of no less than 95% of the precision of the hybrid method implemented with a full graph traversal while reducing processing time. It is crucial to clarify the distinction between absolute precision and the relative efficiency of the optimization. The '95% precision' claim refers specifically to the relative retention of the search results compared to an exhaustive (full) graph traversal. In other words, by limiting the search area to 30–50 neighboring nodes, the algorithm maintains 95% of the maximum possible context that could be retrieved if the entire graph were searched. While the absolute precision remains at the 0.24–0.25 level due to the inherent complexity of multi-hop logistics queries, this relative metric demonstrates that the proposed optimization successfully achieves high scalability with a negligible (less than 5%) loss of the 'best possible' information available in the knowledge graph.

The observed performance patterns reflect the inherent limitations of each search logic. Vector search is fundamentally prone to failures in multi-hop scenarios where the required information is topologically distant, even if semantically related. Pure graph search, conversely, fails when there is a lexical mismatch between the user's natural language and the database schema. The hybrid method effectively mitigates these issues, with its primary constraints being the predefined traversal depth and neighbor limits. These parameters

represent a controlled trade-off where retrieval may stop before capturing an exhaustive causal chain to maintain real-time system responsiveness.

The experimental results confirm that the hybrid method provides higher precision (up to 0.25) compared to purely vector-based and graph-based approaches. In terms of query processing time, the hybrid algorithm demonstrates scalability: when the data volume increases to around 2000 key nodes, the delay does not exceed 0.20 s, whereas the execution time of pure graph search rises to 3.78 s. The obtained data justify the applicability of the hybrid approach for handling queries in systems with large volumes of logistical data.

6. Conclusions

The article discusses methods of organizing information search in logistics robotic systems. It was proposed to use a hybrid method based on the combined application of a vector-graph data representation structure and large language models, with the aim of improving the efficiency of logistics systems in terms of the precision and speed of processing user requests under conditions of limited computational resources. The results of experiments on synthetic datasets (530 nodes, 43,630 edges; 1060 nodes, 175,010 edges; 2120 nodes, 700,020 edges) showed that the hybrid method achieved precision scores of 0.24–0.25 in all series of experiments, which is the highest among the methods studied. In terms of processing speed, the hybrid method reaches 0.09 s (the logistic structure contains 530 nodes and 43,630 edges), 0.13 s (the logistic structure contains 1060 nodes and 175,010 edges), and 0.20 s (the logistic structure contains 2120 nodes and 700,020 edges), and is second only to the pure vector method.

Thus, it can be concluded that from the perspective of query processing speed, the hybrid method demonstrates resilience to scaling. The range of the number of neighboring nodes to be traversed when implementing the hybrid method was also determined, which ensures achieving a relative precision retention of at least 95% compared to a full graph traversal while reducing processing time, corresponding to a range of 30–50 nodes. Therefore, the experiments showed that combining the vector method for the initial selection of relevant nodes and the graph method for restoring connectivity produces a synergistic effect, critically important for handling complex multi-component queries, ensuring the high efficiency of the hybrid method for processing complex user queries in natural language in logistics systems.

Despite the promising results, this study has certain limitations regarding its external validity. The evaluation was conducted on synthetic datasets, which, while accurately reflecting logistics scenarios, may not fully capture the linguistic diversity and “noise” present in real-world documentation. Additionally, the identified optimal hyperparameters (30–50 nodes) are specific to the tested graph scales and may require further adjustment for significantly larger or more fragmented network structures.

Future research will focus on validating the proposed hybrid method using real-world logistics datasets from industrial partners to test its robustness against the complexities of physical systems, including noisy sensor data, incomplete information, and contradictory updates. Further testing will involve a wider variety of graph topologies and authentic queries characterized by high linguistic variability. Refinement of the ranking algorithm is also planned through investigation of adaptive weighting for the semantic and structural components (alpha and beta parameters) to improve precision across different types of logistics tasks. Finally, the generalizability of this approach will be evaluated by integrating it with different base large language models.

The proposed method and findings are of practical value to developers of intelligent logistics platforms, as well as to supply chain managers requiring precise and scalable natural language interfaces for handling dynamic data in real-time operational

environments. Furthermore, the identified optimal graph traversal parameters provide a practical benchmark for balancing computational efficiency and retrieval accuracy in resource-constrained systems.

Author Contributions: Conceptualization, V.V., Y.M., I.S., E.L., O.K. and A.K.; methodology, V.V., Y.M. and E.L.; software, V.V., Y.M. and E.L.; validation, I.S., O.K. and A.K.; formal analysis, V.V., Y.M. and E.L.; investigation, I.S.; resources, A.K.; data curation, O.K.; writing—original draft preparation, V.V., Y.M. and I.S.; writing—review and editing, O.K.; visualization, A.K.; supervision, O.K.; project administration, O.K.; funding acquisition, O.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is available upon request.

Acknowledgments: This research was funded by the Strategic Academic Leadership Program of the Southern Federal University ("Priority 2030").

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
LLM	Large Language Model
RAG	Retrieval-Augmented Generation
DBMS	Database Management System
IoT	Internet of Things
IIoT	Industrial Internet of Things
ERP	Enterprise Resource Planning
WMS	Warehouse Management System
TF-IDF	Term Frequency–Inverse Document Frequency
DPR	Dense Passage Retrieval
SBERT	Sentence-BERT
NLP	Natural Language Processing
BERT	Bidirectional Encoder Representations from Transformers
RoBERTa	Robustly Optimized BERT Pre-training Approach
ALBERT	A Lite BERT
BART	Bidirectional and Auto-Regressive Transformer
T5	Text-to-Text Transfer Transformer
FAISS	Facebook AI Similarity Search
BFS	Breadth-First Search
PPR	Personalized PageRank

References

1. Parkhomenko, V.; Gayda, T.; Medvedev, M. Intelligent System for Countering Groups of Robots Based on Reinforcement Learning Technologies. In *Frontiers in Robotics and Electromechanics. Smart Innovation, Systems and Technologies*; Ronzhin, A., Pshikhopov, V., Eds.; Springer: Singapore, 2023; Volume 329, pp. 135–146. [CrossRef]
2. Klimenko, A.; Safronenkova, I. Efficiency of Data Preprocessing Application for Geographically Distributed Cyberphysical Systems Using WSNs. In *Proceedings of the 2024 International Russian Automation Conference (RusAutoCon), Sochi, Russia, 8–14 September 2024*; IEEE: Sochi, Russia, 2024; pp. 1200–1204. [CrossRef]

3. Sayfullaev, D.; Balsem, Z.A.; Majeed, A.; Gulomjonugli, A.; Bipin Shivaji, A.A. Autonomous Logistics: The Role of Artificial Intelligence, Drones, and Robotics in Supply Chain Management. In *Proceedings of the 2025 International Conference on Computational Innovations and Engineering Sustainability (ICCIES), Ho Chi Minh City, Vietnam, 23–25 July 2025*; IEEE: Coimbatore, India, 2025; pp. 1–6. [CrossRef]
4. Kavre, M.; Gadekar, A.; Gadhade, Y. Internet of Things (IoT): A Survey. In *Proceedings of the 2019 IEEE Pune Section International Conference (PuneCon), Pune, India, 18–20 December 2019*; IEEE: Pune, India, 2019. [CrossRef]
5. Medvedev, M.; Pshikhopov, V. Path Planning of Mobile Robot Group Based on Neural Networks. In *Lecture Notes in Computer Science*; Springer International Publishing: Cham, Switzerland, 2020; pp. 51–62. [CrossRef]
6. Melnik, E.; Safronenkova, I. A Method Based on a Combination of Ontological Data Analysis and Cognitive Modeling Tools for Organizing the Computational Process in Peer-to-Peer Networks. In *Proceedings of the 2024 International Russian Automation Conference (RusAutoCon), Sochi, Russia, 8–14 September 2024*; IEEE: Sochi, Russia, 2024; pp. 854–860. [CrossRef]
7. Dhanwe, S.; Chopade, M.A.; Kazi, K.; Liyakat, S. AI-Driven IoT in Robotics: A Review. *J. Mech. Robot.* **2024**, *9*, 41–48.
8. Pal, D.K.D.; Chitta, S.; Bonam, V.S.M.; Katari, P.; Thota, S. AI-Assisted Project Management: Enhancing Decision-Making and Forecasting. *J. Artif. Intell. Res.* **2023**, *3*, 146–171.
9. Zelenkov, Y.A. Knowledge Management in Organization and the Large Language Models. *Russ. Manag. J.* **2024**, *22*, 573–601. [CrossRef]
10. Raiaan, M.A.K.; Mukta, M.S.H.; Fatema, K.; Fahad, N.M.; Sakib, S.; Mim, M.M.J.; Ahmad, J.; Ali, M.E.; Azam, S. A Review on Large Language Models: Architectures, Applications, Taxonomies, Open Issues and Challenges. *IEEE Access* **2024**, *12*, 26839–26874. [CrossRef]
11. Peng, B.; Zhu, Y.; Liu, Y.; Bo, X.; Shi, H.; Hong, C.; Zhang, Y.; Tang, S. Graph Retrieval-Augmented Generation: A Survey. *ACM Trans. Inf. Syst.* **2025**, *44*, 1–52. [CrossRef]
12. Evtodieva, T. Evolution of Logistics Management: From System to Ecosystem. *Bull. South Ural State Univ. Ser. Econ. Manag.* **2025**, *19*, 186–194. [CrossRef]
13. Koll, S. AI Boost for Logistics: DHL Leads While Others Struggle. Available online: https://trans.info/en/ai-study-dhl-442015 (accessed on 8 January 2026).
14. How Amazon Robots Navigate Congestion. Available online: https://www.amazon.science/latest-news/how-amazon-robots-navigate-congestion (accessed on 8 January 2026).
15. Walmart Reveals Plan for Scaling Artificial Intelligence, Generative AI, Augmented Reality and Immersive Commerce Experiences. Available online: https://corporate.walmart.com/news/2024/10/09/walmart-reveals-plan-for-scaling-artificial-intelligence-generative-ai-augmented-reality-and-immersive-commerce-experiences (accessed on 8 January 2026).
16. DeNittis, N. Artificial Intelligence at Procter & Gamble. Available online: https://emerj.com/artificial-intelligence-at-procter-gamble/ (accessed on 8 January 2026).
17. ManualsLib—Makes It Easy to Find Manuals Online! Available online: https://www.manualslib.com/products/Honeywell-Vocollect-10731939.html (accessed on 8 January 2026).
18. Advancing Warehouse Management with Voice Picking Software. Available online: https://www.lucasware.com/voice-directed-warehousing/ (accessed on 8 January 2026).
19. Supply Chain Compass Blog. Available online: https://blog.blueyonder.com/luminate-platform-built-on-three-guiding-principles/ (accessed on 8 January 2026).
20. Bianchini, F. Retrieval-Augmented Generation. In *Engineering Information Systems with Large Language Models*; Springer Nature: Cham, Switzerland, 2025. [CrossRef]
21. Rocchio, J. *Relevance Feedback in Information Retrieval*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1971.
22. Lv, C. Adaptive Term Frequency Normalization for BM25. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management, Glasgow, UK, 24–28 October 2011*; Association for Computing Machinery: New York, NY, USA, 1985.
23. Roussel, R.; Edelen, A.L.; Boltz, T.; Kennedy, D.; Zhang, Z.; Ji, F.; Huang, X.; Ratner, D.; Garcia, A.S.; Xu, C.; et al. Bayesian Optimization Algorithms for Accelerator Physics. *Phys. Rev. Accel. Beams* **2024**, *27*, 084801. [CrossRef]
24. Géry, M.; Largeron, C. BM25t: A BM25 Extension for Focused Information Retrieval. *Knowl. Inf. Syst.* **2012**, *32*, 217–241. [CrossRef]
25. Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; Yih, W.-T. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Virtual Meeting, 16–20 November 2020*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020.
26. Luo, K.; Qin, M.; Liu, Z.; Xiao, S.; Zhao, J.; Liu, K. Large Language Models as Foundations for Next-Gen Dense Retrieval: A Comprehensive Empirical Assessment. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, Miami, FL, USA, 12–16 November 2024*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2024; pp. 1354–1365. [CrossRef]
27. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-T.; Rocktäschel, T.; et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9459–9474.

28. Moon, A.-S.; Kim, K.; Lee, J. Data Augmentation for Dense Passage Retrieval Using Corpus-Passage Frequency-Based Token Deletion. *J. Big Data* **2025**, *12*, 199. [CrossRef]
29. He, G.; Lin, C.; Ren, J.; Duan, P. Predicting the emergence of disruptive technologies by comparing with references via soft prompt-aware shared BERT. *J. Informetr.* **2024**, *18*, 101596. [CrossRef]
30. Wang, X.; Macdonald, C.; Tonellotto, N.; Ounis, I. ColBERT-PRF: Semantic Pseudo-Relevance Feedback for Dense Passage and Document Retrieval. *ACM Trans. Web* **2022**, *17*, 1–39. [CrossRef]
31. Rahman, M.H.; Uddin, M.A.; Ria, Z.F.; Rahman, R.M. Optimizing BERT for Bengali Emotion Classification: Evaluating Knowledge Distillation, Pruning, and Quantization. *Comput. Model. Eng. Sci.* **2025**, *142*, 1637–1666. [CrossRef]
32. Aparna, B.S.; Remya, S.; Pillai, M.J.; Subbareddy, S.R.; Cho, Y.Y. ALBERT-BiLSTM Cross-Attention Network with Progressive Knowledge Distillation for Multi-Domain SMS Spam Classification. *Results Eng.* **2025**, *27*, 106727. [CrossRef]
33. Relan, S.; Rambola, R.K. Big Bird-Disentangled Representation and Adaptive Contrast Transformer for Abstractive Text Summarization with Contrast Attention. *Eng. Appl. Artif. Intell.* **2025**, *162*, 112536. [CrossRef]
34. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Seattle, WA, USA, 5–10 July 2020*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020.
35. Rodriguez-Torrealba, R.; Garcia-Lopez, E.; Garcia-Cabot, A. End-to-End Generation of Multiple-Choice Questions Using Text-to-Text Transfer Transformer Models. *Expert Syst. Appl.* **2022**, *208*, 118258. [CrossRef]
36. Heise, I.; Esser, S.; Borrmann, A. A Graph-Based Systems-of-Systems Architecture Enabling Multi-Scale Digital Twins for Maintaining Road Infrastructure. *Adv. Eng. Inform.* **2025**, *68*, 103649. [CrossRef]
37. Kazakova, E.D. Analysis of the Modern Algorithms' Accuracy for Communities Identification on Networks When Working with Graph Databases. *Comput. Nanotechnol.* **2023**, *10*, 49–59. [CrossRef]
38. Shalagin, N. A Survey on Natural Language Semantic Search Algorithms. *Int. J. Open Inf. Technol.* **2024**, *12*, 11–21.
39. Oleynik, A.; Fedorov, A.; Datyev, I.; Zuenko, A.; Shestakov, A.; Vishnyakov, I. Using Rag Technology to Design an Intel-Ligent Information System for Support Exploratory Search. *Trans. Kola Sci. Cent.* **2025**, *15*, 5–26. [CrossRef]
40. Kozlovsky, A.V.; Melnik, Y.E.; Voloshchuk, V.I. An Approach for Making a Conversation with an Intelligent Assistant. In *Lecture Notes in Networks and Systems*; Springer International Publishing: Cham, Switzerland, 2023; pp. 526–536. [CrossRef]
41. Voloshchuk, V.I.; Garyagdiyev, A.; Kozlovskaya, M.A.; Melnik, Y.E.; Samoylov, A.N. An Approach to Building Adaptive Object Accounting Systems Using Artificial Intelligence Methods. *IZV. SFedU Eng. Sci.* **2024**, *5*, 152–162. [CrossRef]
42. Kozlovskiy, A.V.; Melnik, E.V.; Samoylov, A.N. A Large Language Models Application in Organization of Replenishment of the Knowledge Base on Methods of Information Processing in Systems of Applied Photogrammetry. *IZV. SFedU Eng. Sci.* **2024**, *239*, 243–252. [CrossRef]
43. Introduction to Beam Search Algorithm. Available online: https://www.geeksforgeeks.org/machine-learning/introduction-to-beam-search-algorithm/ (accessed on 8 January 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.