Doing More with Less: A Survey on Routing Strategies for Resource Optimisation in Large Language Model-Based Systems

CLOVIS VARANGOT-REILLE*, Wikit, France and Laboratoire Hubert Curien, UMR CNRS 5516, France
CHRISTOPHE BOUVARD, Wikit, France
MATHIEU CIANCONE, Wikit, France
ANTOINE GOURRU, Laboratoire Hubert Curien, UMR CNRS 5516, France
MARION SCHAEFFER, INSA Rouen Normandie, France
FRANÇOIS JACQUENET, Laboratoire Hubert Curien, UMR CNRS 5516, France

Background: Large Language Model (LLM)-based systems, such as conversational agents, are usually designed with monolithic, static architectures that rely on a single, general-purpose LLM to handle all user queries. However, these systems may be inefficient as different queries may require different levels of reasoning, domain knowledge or pre-processing. While generalist LLMs (e.g. GPT-4o, Claude-Sonnet) perform well across a wide range of tasks, they may incur significant financial, energy and computational costs. These costs may be disproportionate for simpler queries, resulting in unnecessary resource utilisation. A routing mechanism can therefore be employed to route queries to more appropriate components, such as smaller or specialised models, thereby improving efficiency and optimising resource consumption.
Objectives: This survey aims to provide a comprehensive overview of routing strategies in LLM-based systems. Specifically, it reviews when, why, and how routing should be integrated into LLM pipelines to improve efficiency, scalability, and performance.
Methods: We structure the survey by defining the objectives to optimise, such as cost minimisation and performance maximisation; the timing of routing within the LLM workflow, whether it occurs before or after generation; and the various implementation strategies, including similarity-based, supervised, reinforcement learning-based, and generative methods. Practical considerations such as industrial applications and current limitations are also examined, like standardising routing experiments, accounting for non-financial costs, and designing adaptive strategies.
Results: There is a wide range of routing strategies, from lightweight, similarity-based and supervised methods, to more complex approaches involving LLM fine-tuning and reinforcement learning. Most current strategies adopt a pre-generation approach, which is generally more resource-efficient. This survey demonstrates that some low-resource solutions can provide generalisation capabilities.
Conclusions: Routing offers a practical way to improve the efficiency of LLM-based systems. By formalising routing as a performance–cost optimisation problem, this survey provides tools and directions to guide future research and development of adaptive low-cost LLM-based systems.

JAIR Track: AI and Society Track

JAIR Associate Editor: Bo Han

*Corresponding author. All of the authors listed after the first one contributed equally to this work.

Authors' Contact Information: Clovis Varangot-Reille, ORCID: 0000-0002-4188-8013, clovis.varangot@wikit.ai, Wikit, Lyon, France and Laboratoire Hubert Curien, UMR CNRS 5516, Saint-Etienne, France; Christophe Bouvard, ORCID: 0009-0006-7962-0270, christophe.bouvard@wikit.ai, Wikit, Lyon, France; Mathieu Ciancone, ORCID: 0009-0003-6607-9583, mathieu.ciancone@wikit.ai, Wikit, Lyon, France; Antoine Gourru, ORCID: 0000-0003-3571-2430, antoine.gourru@univ-st-etienne.fr, Laboratoire Hubert Curien, UMR CNRS 5516, Saint-Etienne, France; Marion Schaeffer, ORCID: 0000-0001-9854-7482, marion.schaeffer@insa-rouen.fr, INSA Rouen Normandie, Rouen, France; François Jacquenet, francois.jacquenet@univ-st-etienne.fr, ORCID: 0000-0002-0653-0710, Laboratoire Hubert Curien, UMR CNRS 5516, Saint-Etienne, France.

DOI: 10.1613/jair.1.19801

JAIR Reference Format:
Clovis Varangot-Reille, Christophe Bouvard, Mathieu Ciancone, Antoine Gourru, Marion Schaeffer, and François Jacquenet. 2026. Doing More with Less: A Survey on Routing Strategies for Resource Optimisation in Large Language Model-Based Systems. *Journal of Artificial Intelligence Research* 86, Article 14 (June 2026), 32 pages. DOI: 10.1613/jair.1.19801

1 Background

In Large Language Model (LLM)-based systems, a router is a component that receives text data (e.g., a user query) and decides which candidate (e.g. LLM, workflow) in a pool should be invoked next in order to process the text or answer the user's query. It serves as a system orchestration component. The aim of most existing routing approaches is to optimise resource consumption by finding the most cost-effective model (see Section 2 for more details).

While this concept shares the concept of conditional selection with the Mixture-of-Experts (MoE) framework (Jacobs et al. 1991), where multiple "experts" learn to handle different parts of the input space, the two paradigms address fundamentally different problems and operate under distinct constraints.

In traditional MoE architectures, an internal gating mechanism, generally a linear layer with a softmax activation, selects a subset of expert subnetworks to process each input (Cai et al. 2025; Fedus et al. 2022). The primary objective is to increase model capacity and specialization while maintaining computational efficiency during inference through sparse activation. Experts are trained end-to-end together with the gating function, enabling the system to learn implicit task decomposition. However, this connection limits MoE to scenarios where: (i) all components are neural networks; (ii) experts can be jointly optimized; and (iii) the expert pool remains fixed after training.

In contrast, routing in LLM-based systems operates as an external orchestration layer that addresses deployment challenges beyond model architecture. Routing in LLM-based system: (i) operates externally and independently of the LLM internal forward process, enabling routing decisions based on non-differentiable criteria such as cost, latency, or API availability; (ii) operates under explicit budget constraints when accessing external candidates, requiring trade-offs between performance and resource consumption (e.g., using a smaller model when acceptable to reduce costs); and (iii) supports heterogeneous candidates that are not limited to neural networks and can evolve over time, like LLM with different capabilities, specialized tools, retrieval systems, or entire workflows.

From a high-level perspective where routing is defined as a selection function, MoE could be interpreted as one particular implementation, namely *intra-model routing*, while LLM-based system routers perform *inter-component routing*. However, treating these paradigms as equivalent obscures fundamental differences in their objectives, constraints, and practical applications.

Overall, routing in LLM-based systems involves the orchestration of pre-existing independent systems. This pool of systems may evolve over time and have variable costs and availability, serving distinct functional roles. This makes the typical candidate pool environment highly dynamic, requiring tailored methods to solve the general problem.

In a more formal way, a router is defined as follows. Given a set of n models, and more generally routing candidates, $\mathcal{M} = \{M_1, ..., M_n\}$, for a given query q, the router function $\mathcal{R}$ aims to maximise the scoring function s (see Section 2.1 for scoring function examples) while adhering to a budget constraint B:

$$\begin{aligned} \mathcal{R}_{\mathcal{M}}(q) &= \arg\max_{M \in \mathcal{M}} s(q, M) \\ &\text{s.t. } C_M(q) \leq B \end{aligned} \tag{1}$$

where $C_M(q)$ is a cost function that maps a query-model pair to a non-negative value representing the resource consumption for processing query q with model M (see Section 2.2 for cost function examples) and B is the budget constraint representing the maximum amount of resources available. When $B = +\infty$, there is no budget

constraint, and the routing system selects the model that maximises the scoring function.
For scenarios requiring a more balanced trade-off between performance and cost (Q. J. Hu et al. 2024; Jitkrittum et al. 2025), Equation 1 can be reformulated to incorporate cost directly into the objective function:

$$\begin{aligned} \mathcal{R}_{\mathcal{M}}(q) = & \arg\max_{M \in \mathcal{M}} s(q, M) - \lambda C_M(q) \\ & \text{s.t. } C_M(q) \leq B \end{aligned} \tag{2}$$

where λ is a hyperparameter that controls the relative importance of cost versus performance in the routing decision.

This article focuses on one of the most common applications of LLM-based systems: *conversational agents* (Dam et al. 2024). Conversational agents respond to user queries by simulating human conversations (Caldarini et al. 2022; C.-C. Lin et al. 2023). They process users queries and provide relevant answers (Caldarini et al. 2022). The Retrieval-Augmented Generation (RAG) architecture improves the relevance of the responses by implementing an information retrieval step to the process (Y. Gao, Y. Xiong, X. Gao, et al. 2024; P. Lewis et al. 2020), retrieving content from a knowledge base to include alongside the user query in the model prompt. Even compared to fine-tuning, this was shown to reduce hallucinations (Bouvard et al. 2024). A straightforward optimisation of conversational agent architectures is selecting the appropriate pre-trained LLM to answer a query. Models differ significantly in parameter count (e.g., *Llama-3.2-3B* vs. *Llama-3.1-405B*) and domain specialisation (e.g., *mathstral-7B-v0.1*[1] for mathematical tasks or *Med-Palm*[2] for medical-related tasks). Nevertheless, most systems still rely on a single generalist LLM to handle all queries. Routing each query to the most efficient LLM can improve cost-efficiency and response quality by leveraging domain-specific expertise. This prevents using models that are unnecessarily large, have insufficient reasoning capabilities, or lack the required domain knowledge.

Through this work, we conduct a state-of-the-art of the routing frameworks implemented in LLM-based systems. Previous works have addressed routing within broader surveys on LLM inference optimization (Y. Chen et al. 2025; Z. Chen et al. 2025; C. Wang et al. 2024), while Behera et al. (2025) provided an overview of routing frameworks with limited strategy coverage. Our survey offers an in-depth analysis, defines the fundamental components of routing frameworks and introduces a taxonomy that distinguishes pre-generation routing (predicting candidate performance) from post-generation or cascade routing (evaluating a candidate based on its output). We provide an extensive coverage of the routing literature, addressing three core questions (Figure 1):

- *Q1: What should the routing optimise?*
- *Q2: When should routing take place?*
- *Q3: How routing should be implemented?*

Then, we examine key challenges in LLM-based routing systems, including implementation costs, data requirements, generalization to unseen candidates, pool management, and adversarial robustness. We further discuss future research directions: the establishment of standardized benchmarks, consistent baseline comparisons, greater diversity in cost functions, and investigation of candidate complementarity.
We exclude approaches that select the best answer from the answers generated by potential routing candidates (Guha et al. 2024; Si et al. 2023) or ensemble methods that merge outputs from multiple models (J. Hu et al. 2024; D. Jiang et al. 2023; H. Wang et al. 2024). While these strategies can be effective, they prioritise performance over cost efficiency and thus fall outside our definition of routing.

[1] MistralAI's Mathstral
[2] Google's Med-Palm

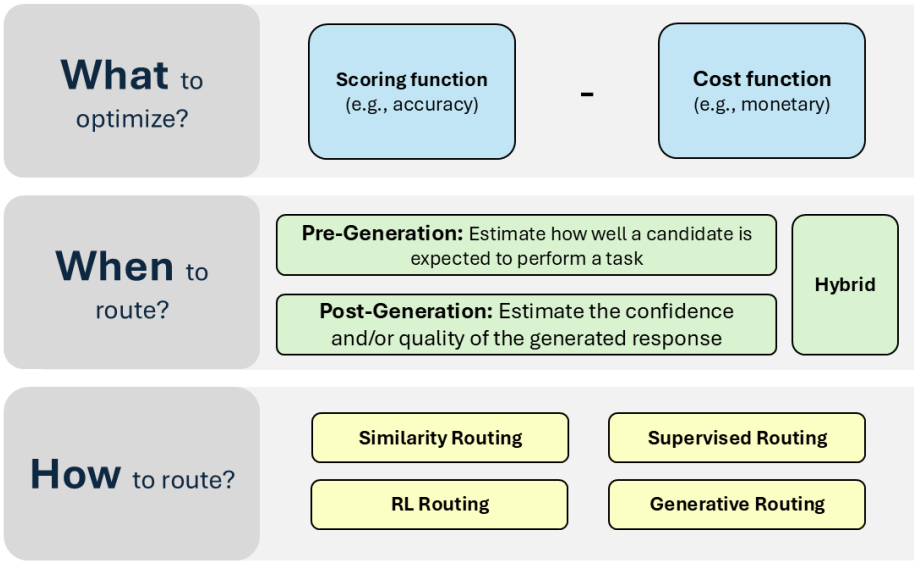

Fig. 1. The flow-chart summarizes key decisions in routing: what to optimize (a scoring function such as accuracy and a cost function such as monetary cost or inference latency), when to route (pre-generation, post-generation, or hybrid approaches), and how to route (similarity-based, supervised, reinforcement learning–based, or generative routing methods).

This survey is structured into several sections that focus on the critical aspects of routing. We begin by describing the essential elements of routing. Next, we examine the pipeline stage at which routing is implemented in the literature. Finally, we detail and classify routing strategies according to the frameworks used. We discuss these strategies considering industrial practices and highlight the key challenges and futures directions that the field must address.

2 Q1: What Should the Routing Optimise?

The primary goal of an efficient routing system is to minimise unnecessary resource consumption and maximise performance by using the most suitable model or system component for the task. In essence, efficient routing aims to optimise the trade-off between performance and cost as formalised in Equation 1 and 2.

2.1 A Performance Metric to Maximise

A key objective of an efficient router function is to maximise a scoring function $s(q, M)$ that evaluates a model's ability to generate accurate answers.

There are several possible scoring function to maximise. The evaluation process in a traditional supervised learning framework involves comparing the generated answers with the ground truth. In cases where data lack annotation, including a human evaluator in the evaluation loop enables to assess whether the response is factually correct, in the expected format, and consistent with the expected ground truth (Chang et al. 2024). However, evaluating thousands of queries demands substantial time and intensive human effort, affecting scalability. In addition, subjective bias might appear while evaluating, such as lack of expertise or preferences. More straightforward strategies include exact matching, partial matching, ROUGE score (C.-Y. Lin 2004), or even

semantic similarity to the ground truth (Chang et al. 2024; L. Zhang et al. 2024). However, these metrics often fall short in evaluating factual accuracy.

To address this, prior research has explored automatic evaluation methods using LLMs. These involve embedding both the rating criteria and the generated response within the prompt (Chiang and H.-y. Lee 2023). While promising, their alignment with human judgment remains uncertain, and outcomes are sensitive to instruction clarity (H. Wei et al. 2024). Many frameworks, such as RAGAS (Es et al. 2024), have been proposed.
Preference learning, based on the preferences of human reviewers, can be used to assess the quality of responses in addition to traditional methods (R. Jiang et al. 2024). Data related to preferences are typically represented as $A \prec_x B$, indicating that for a given query x, the user prefers the generated answer A over the alternative B (Fürnkranz and Hüllermeier 2012).

2.2 A Cost to Minimise

The cost function $C_M(q)$ represents the resource consumption associated with using model M to process query q. This cost dimension can take multiple forms depending on operational priorities.
Most existing LLM-based systems depend on API calls to proprietary models, such as those provided for example by OpenAI, Google, Anthropic, Perplexity, xAI, etc. In this context, the primary cost a router seeks to minimise is the price per token. To accurately estimate the total cost of running a pipeline, one must account for all pre-trained models invoked during the process, including both LLMs and embedding models in RAG settings. Other production-related costs, such as latency or computational costs, can also be minimised (Irugalbandara et al. 2024). Latency is the delay between the user request and the pipeline response. Effective routing could reduce the average latency by routing simple user queries to LLMs that require fewer computing resources.
Beyond monetary and latency costs, we can also consider minimising the environmental cost of an LLM-based system. As LLMs increase in usage and scale, their power and computing requirements grow significantly (Luccioni et al. 2024), growing the ecological impacts associated with LLM-based systems. This impact is often measured considering the application's energy consumption (kWh) or global warming potential (kgCO2eq). Various tools[3] have been proposed to estimate this environmental footprint.

3 Q2: When Should Routing Take Place?

We identify two key stages in the pipeline where routing may take place: *pre-generation routing*, presented in Figure 2, and *post-generation routing* (also known as *cascade routing*), presented in Figure 3. Pre-generation routing occurs before generating a response to the user query, while post-generation routing occurs after generating the response.

3.1 Routing as a Pre-generation Step

In **pre-generation routing**, the system estimates in advance whether an LLM is likely to provide an adequate answer (see Figure 2). This approach minimises latency, as no model needs to generate a response before routing decisions are made. There are two main approaches to achieve this, depending on our use case and LLM setup:

- infer the domain of knowledge of the query and route it to the associated LLMs trained as domain experts
- assess the LLM candidate's ability to answer a query of a given complexity and then route the query to the LLM with sufficient reasoning ability.

In this survey, we define the complexity of a query as an estimate of the degree of difficulty that an LLM is likely to encounter in the generation of a response to a given query. Within the context of RAG-based conversational agents, for example, query complexity can span a spectrum that includes: simple queries requiring no retrieval (such as greetings), queries that involve extracting explicit information from limited document sources, and

[3] Ecologits, CodeCarbon, MLCO2, Boavizta

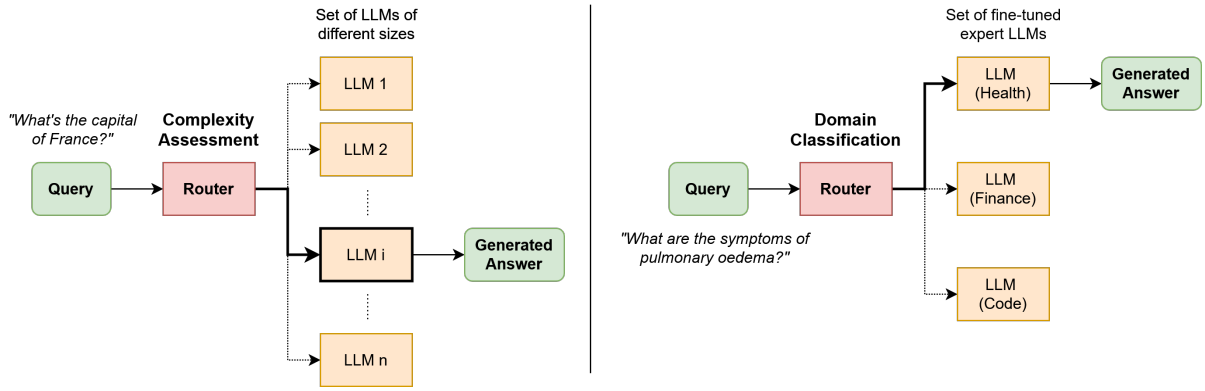

Fig. 2. **Routing as Pre-Generation Step** – Before generating an answer, each LLM's ability to provide an appropriate answer is assessed based on the user query complexity and/or topic. *Dotted arrows represent non-selected LLM candidates. Rectangles with straight edges represent the router and routing candidates, while rectangles with rounded corners represent input/output components (user requests and results).*

queries that require the extraction of implicit information from multiple documents through reasoning. This spectrum reflects the varying requirements of the retrieval and generation components, ranging from simple information retrieval to multi-document reasoning tasks.

3.2 Routing as a Post-Generation Step

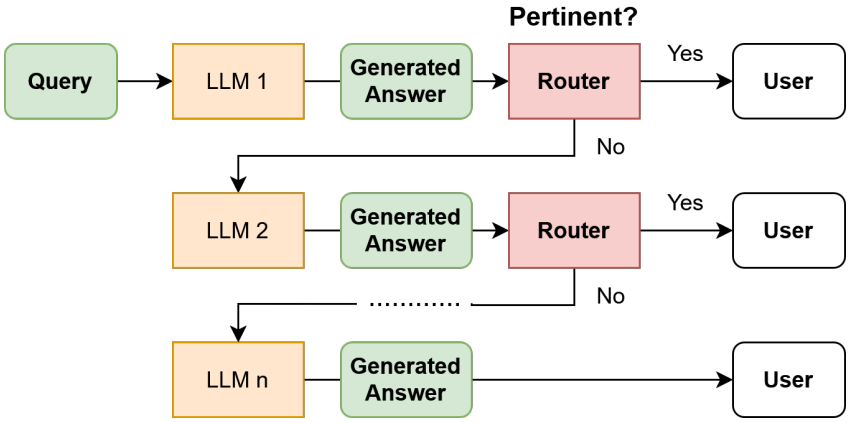

Fig. 3. **Routing as Post-Generation Step (or cascade routing)** – The relevance of a larger LLM is determined by the evaluation of the answers generated by the current LLM. Each candidate response is evaluated sequentially. If an answer is deemed inadequate or untrustworthy, the user query is routed to a larger LLM. Typically, the cascade sequence is static. *Rectangles with straight edges represent the router and routing candidates, while rectangles with rounded corners represent input/output components.*

One might conceptualise routing as a **post-generation step**, where each model response is assessed iteratively (or in a cascade manner) by selecting progressively more advanced models until the response is considered

adequate (see Figure 3). Unlike pre-generation routing, cascade routing evaluates generated responses. (Kamalloo et al. 2023). This method can incur higher latency and cost, since multiple LLMs may need to generate answers for a single query.

A more flexible variant of this approach is the **multi-choice cascading** method (Dekoninck et al. 2025). Although it is still a post-generation technique, it incorporates elements of pre-generation routing by allowing the next model to be selected dynamically. Rather than following a fixed sequence, the router can forward the query to any available LLM based on an evaluation of the current response. While this hybrid strategy generally outperforms simple cascading and traditional pre-generation methods, it still requires multiple generations per query, which impacts cost and latency.

4 Q3: How Routing Should Be Implemented?

This section summarises the various strategies for implementing routing in LLM-based systems, as summarised in Table 1. We group the methods into four main categories : similarity-based, supervised, reinforcement learning-based, and generative routing. The complexity, resource requirements, generalisation and effectiveness of these strategies differ, offering distinct trade-offs between performance, cost and adaptability.

4.1 Similarity-based Routing

Similarity-based routing routes queries to LLMs based on semantically similar queries from past interactions, using unsupervised or weakly supervised signals. This approach includes techniques based on query similarity, clustering of previous interactions, and human preference similarity. While generally lightweight and adaptable, these methods often face limitations when handling complex queries or generalising across diverse tasks.

4.1.1 Query Similarity. This is typically achieved by computing the cosine similarity between query embeddings. The fundamental premise is that similar queries should be processed by the same model. The simplest implementations use 1-Nearest-Neighbour routing (Stripelis et al. 2024), yet these approaches fail to capture complex query-response relationships and often perform worse than random baselines. Leveraging multiple nearest neighbours address this limitation (Jang et al. 2023; Manias et al. 2024)[4] by performing similarly to prompt-based intent detection achieving around 90% accuracy (Manias et al. 2024), while Jang et al. (2023) demonstrated strong performance on non-generative tasks but inferior results on generative tasks. This disparity in performance across different types of task reveals that the effectiveness of routing based on query similarity is highly dependent on the type of task and may be influenced by the complexity of the query.
In the domain of text-to-SQL generation, Malekpour et al. (2024) proposed selecting the cheapest LLM that exceeds a success threshold, defined as the minimum proportion of similar queries for which the LLM returned the correct SQL query. Their method achieved near-optimal performance of 60.1%, compared to 61.0% for the best-performing standalone model (*GPT-4o*), while reducing costs by a factor of 1.1. This modest reduction reflects the predominant routing of queries to *GPT-4o* (81%), which raises concerns about cost-effectiveness: a routing strategy must avoid achieving higher quality by exclusively routing to the larger LLM. Moreover, the routing strategy did not include a specialist code generation model, which could potentially have offered similar quality at lower cost.

The performance of similarity-based routing can be further improved by optimising query representation through contrastive learning (S. Chen et al. 2024; C.-H. Lee et al. 2024). This technique ensures that queries with similar meanings are embedded close together in the representation space, while queries with different

[4] aurelio-labs/semantic-router

Table 1. Algorithm-based taxonomy of routing strategies with their generalisation to new candidates and implementation complexity characteristics. *RL: Reinforcement Learning; perf.: performance; Inf.: Inference.* * Studies on Query Complexity Inf.: (Dekoninck et al. 2025; Ding et al. 2024; Q. J. Hu et al. 2024; Jeong et al. 2024; S. Lee et al. 2025; Malekpour et al. 2024; Ning et al. 2024; Ong et al. 2025; Shnitzer et al. 2024; Somerstep et al. 2025; Srivatsa et al. 2024; Stripelis et al. 2024; X. Wang et al. 2025; T. Zhang et al. 2025)

Strategies (Studies)	***Generalisation to New Candidates***	***Implementation Complexity***
Similarity Routing		
Query Similarity (S. Chen et al. 2024; Jang et al. 2023; C.-H. Lee et al. 2024; Malekpour et al. 2024; Manias et al. 2024; Srivatsa et al. 2024; Stripelis et al. 2024)	Yes, query-based matching	Low compute, requires query-perf. corpus. More compute if contrastive learning.
Clustering (Jitkrittum et al. 2025; Pichlmeier et al. 2024; Srivatsa et al. 2024)	Yes, cluster-based assignment	Low compute, requires query-perf. corpus
Preference Similarity (Ong et al. 2025; T. Zhang et al. 2025; Z. Zhao, Jin, et al. 2024)	Yes, similarity-weighted preference matching	Low compute, requires preference data corpus
Supervised Routing		
Recommendation System (Ong et al. 2025; T. Zhang et al. 2025; R. Zhuang et al. 2025)	No, need re-training	Medium compute, need preference data
Domain Classification (Jain et al. 2024; Y. Wang et al. 2024)	No, need re-training	Low-medium compute, need static well-defined domains
*Query Complexity Inf.**	No, need re-training	Low-medium compute, requires training on query-perf. pairs
Knowledge Graph (Feng et al. 2025)	Yes, inductive graph learning via node embeddings	Medium compute, requires building/encoding heterogeneous graph and GNN training
Answer Confidence Inf. (L. Chen et al. 2023)	Limited, binary decisions only	Medium compute , requires training on answer-correctness pairs
RL Routing		
Stateless (Sikeridis et al. 2024)	No, need re-training	Low compute, online reward feedback required
State-based (Y. Li 2025; Nguyen et al. 2024)	Yes, via identity vectors	Medium-high compute, requires query corpus and online feedback
Reward Inf. (Hari and Thomson 2023; Lu et al. 2024)	No, need re-training	High compute, requires training reward model on query-loss pairs
Generative Routing		
Token Probability (Ramírez et al. 2024)	Limited, binary decisions only	High Compute, open-source LLM required
Sequence Probability (C.-H. Lee et al. 2024)	Limited, binary decisions only	High compute, open-source LLM required
Prompt-based Routing (Z. Li et al. 2024; Ning et al. 2024; Shen et al. 2023)	Yes, prompt instructions update	High compute, proprietary LLM compatible
LLM Fine-Tuning (Chai et al. 2024; Chuang et al. 2024; J. Liu et al. 2024; Ong et al. 2025; Patil et al. 2024)	Limited, via prompting	High compute, requires fine-tuning infrastructure
Repeated Calls (Aggarwal et al. 2024; Yue et al. 2024)	Limited, binary decisions only	High compute, proprietary LLM compatible, multiple inferences required
Code Execution (J. Zhang et al. 2023)	Limited, binary decisions only	High compute, proprietary LLM compatible, code execution required

meanings are placed further apart (Le-Khac et al. 2020). While this addresses the semantic representation problem, it introduces substantial training overhead and requires domain-specific tuning.
In the context of task-oriented dialogue, where systems must extract user intents and track dialogue context, C.-H. Lee et al. (2024) apply contrastive learning within a framework called *OrchestraLLM*. *OrchestraLLM* shows a slight improvement in accuracy compared to a direct call to the larger LLM while reducing the number of calls to this model by 50% to 80%. Z. Zhao, Gan, et al. (2024) extends contrastive learning-based routing to select the most appropriate LoRA adapter for a given task. With this approach, the authors dynamically switch LLM capabilities instead of deploying multiple LLMs by selecting the most appropriate LoRA adapter. Z. Zhao, Gan, et al. (2024) found that their contrastive learning-based LoRA adapter routing strategy was particularly effective for natural language understanding tasks such as translation or description generation from structured data. For natural language processing tasks such as summarisation and sentiment analysis, their approach showed either superior performance or performance comparable to an average pooling of multiple adapters retrieved using their method. S. Chen et al. (2024) present a more sophisticated approach called *RouterDC*, which applies multi-level contrastive learning to enhance query representation, using multi-level contrastive learning to map both queries and LLMs into shared embedding spaces. Except for the MMLU dataset, *RouterDC* outperforms the best stand-alone LLMs and performs as well as or better than other routing methods, like *Zooter* (Lu et al. 2024) or supervised approaches, on different datasets. It achieves the highest average performance across out-of-distribution datasets by effectively approximating the results of the best-performing models on each dataset.

4.1.2 Clustering Previous Interactions. The clustering approach treats routing as a two-stage process. First, it identifies the semantic cluster that best represents a query. Then, it selects the LLM that has historically performed best for that cluster type.
Multiple implementations using *k-means* clustering have studied this strategy, though with varying degrees of success. The Expert Router framework (Pichlmeier et al. 2024) leverages pre-trained clusters from large-scale datasets, while other approaches (Jitkrittum et al. 2025; Srivatsa et al. 2024) train clusters from scratch. While Pichlmeier et al. (2024) do not specify how LLMs are assigned to clusters, Srivatsa et al. (2024) and Jitkrittum et al. (2025) evaluate which LLM most frequently produces responses matching the ground truth on each cluster.
When Srivatsa et al. (2024) trained a *k-means* model from scratch for the routing task, the resulting clusters failed to generalise well from training to test datasets. This indicates that both the size and diversity of the training set are critical for effective clustering and generalisation on out-of-distribution tasks. The choice of encoding strategy, whether using a dense representation strategy with *RoBERTa* (L. Zhuang et al. 2021) or a sparse strategy with *TF-IDF*, did not significantly affect the results.

Jitkrittum et al. (2025) proposed a generalisable routing strategy which does not require retraining when introducing new LLM candidates. The approach uses *k-means* clustering to form clusters from a dataset. Then, they evaluate LLM performance on the cluster examples. They calculate the average performance of each LLM across every cluster, representing each LLM as a performance vector amongst clusters. During inference, the system routes new queries to the LLM with the lowest error rate for the nearest cluster to the query. This strategy is dynamic in that the cluster-level performance representation is independent of the original pool of candidate LLMs. Introducing a new model only requires computing its per-cluster error vector on the different clusters. This approach is successful in generalising when different LLMs are used for training and testing; the performance remains comparable to setups involving all candidate LLMs. Furthermore, learning a more sophisticated clustering assignment map does not improve upon the unsupervised approach.

4.1.3 Preference Similarity. An alternative approach uses human preference data to predict which model users would favour for similar queries. This strategy redefines routing as a problem of estimating probabilities: given the similarity of a query to previous examples, what is the likelihood that a more expensive model would be preferred? Both strategies used preference data to develop ranking-based algorithms.

Ong et al. (2025) used preference data from the Chatbot Arena Leaderboard [5], a preference-based LLM ranking interface, to propose *RouteLLM*: a range of routing strategies based on user preferences[6] [7]. They reformulate the routing task between a smaller LLM, *Mixtral-8x7B*, versus a larger LLM, *GPT-4-1106-preview*, as predicting the probability that the larger model will be preferred for a given query. To compute this probability, the authors employ a Bradley-Terry (BT) algorithm (Bradley and Terry 1952) with similarity-weighted ranking. They weight the queries from the training dataset according to their similarity to the user query and subsequently use these weighted queries to learn the BT coefficients. These coefficients then allow the estimation of the probability. This strategy is influenced by the embedding strategy as T. Zhang et al. (2025) compared a strategy where they simulate user preferences by exploiting uncertainty through semantic entropy. In their small-vs-large LLM setup, the model with lower semantic entropy is assumed to be preferred. Their enhanced similarity-weighted ranking strategy shows clear improvements over Ong et al. (2025)'s original approach, requiring only 27% of queries to be routed to the larger model on MT-Bench in order to boost the smaller model's performance by 50%.

In parallel, Z. Zhao, Jin, et al. (2024) introduced an ELO-based algorithm called *Eagle* (Elo 1978). This method incorporates a two-level structure: the global ranking is computed using all available pairwise comparisons to establish an overall model hierarchy; the local ranking is based on pairwise preferences among the nearest neighbor queries, determined via cosine similarity. For any given query, Eagle selects the optimal LLM by computing a weighted sum of its global and local ELO scores, while ensuring the chosen model stays within the user's budget constraints. Notably, these preference-based methods consistently outperform traditional classification approaches (e.g., *SVM*, *multilayer perceptron*, *K-NN*, *BERT*-based classifier or LLMs fine-tuned for classification) (Ong et al. 2025; Z. Zhao, Jin, et al. 2024), suggesting that similarity-weighted preferences capture routing requirements better than standard supervised learning. However, the original *RouteLLM* strategy performed only slightly better than random routing (56% vs. 51%) on one of the experiment of Z. Zhao, Jin, et al. (2024). The authors suggest that this modest improvement may result from the inherent unreliability of human judgments in preference data (T. Zhang et al. 2025).
These results highlight the potential of using information about user preferences to train routing algorithms.

4.2 Supervised Routing

Similarity-based routing frequently fails on complex tasks due to its unsupervised nature, particularly when discriminating between similar tasks or when noise levels are substantial. Once LLM performance profiles have been established for different query categories or knowledge domains, routing can be treated as a supervised task.

4.2.1 Recommendation Systems. Recommendation systems align with the routing framework philosophy by aiming to select the most suitable LLM for each specific query. It has been studied matrix factorisation, a well-known algorithm in recommendation systems (Y. Zhang 2022), which infers latent features from user-item interactions to predict preferences (Ong et al. 2025; T. Zhang et al. 2025; R. Zhuang et al. 2025).
This approach extrapolates users preferences between models and queries by learning a hidden scoring function that reflects user preference patterns (Ong et al. 2025). Enhanced implementations incorporate uncertainty signals through semantic entropy (T. Zhang et al. 2025) or reconstruct comprehensive performance matrices across multiple benchmarks (R. Zhuang et al. 2025).
Matrix factorisation proves more efficient than classification-based approaches for cost-quality trade-offs, with implementations achieving 80% quality gains while requiring only 30% of calls to *GPT-4-1106-preview* (Ong et al. 2025). In contrast, the random assignment required 78% of calls to this model. However, the effectiveness of this approach varies significantly depending on the implementation and evaluation conditions. Some studies have

[5] Chatbot Arena Leaderboard
[6] lm-sys/RouteLLM
[7] https://huggingface.co/routellm

found that it achieves only minimal improvement over random routing (T. Zhang et al. 2025). T. Zhang et al. (2025) improved performance by incorporating uncertainty signals into the embedding inputs. R. Zhuang et al. (2025) confirmed these findings, demonstrating that this strategy approaches the performance of an Oracle router while scaling to over 100 candidate models.

4.2.2 Domain Classification. Domain-based routing routes queries to specialized models based on their knowledge domain, such as health, mathematics, coding, etc. This strategy focuses on broad categorical knowledge rather than query-specific details.

BERT-based implementations, like fine-tuned *DeBERTA-v3-large* model (He et al. 2023a; Simonds et al. 2024) demonstrate the effectiveness of this approach (Simonds et al. 2024; Y. Wang et al. 2024), with systems like *MoDEM* achieving accuracy comparable to that of larger, generalist, stand-alone LLMs, while routing to domain-specific expert models (Simonds et al. 2024), but at lower costs. Simonds et al. (2024)'s approach achieved an MMLU accuracy of 87.7%, which is closer to that of *Llama-3.1-405B* (Grattafiori et al. 2024) (88.6%) than *Qwen 2.5-72B* (Yang et al. 2024) (86.1%), while using a pool of expert 70B models at a cost per token four times lower than that of the 405B model (Simonds et al. 2024).

BERT-based models are not strictly necessary for effective domain-based routing. Jain et al. (2024) proposed a two-stage routing strategy called *Composition-of-Experts*. In the first step, a *k-NN* model maps queries to knowledge domains. If the entropy-based measure of classifier uncertainty exceeds a predefined threshold, the model assigns the user query to the "general" category. In the second step, a mixed integer linear program allocates categories to experts, aiming to minimise costs while adhering to budget constraints. For certain datasets, such as Arena-Hard or MT-Bench, this framework achieved scores comparable to LLMs, like *Llama-3-70B-Instruct* (Grattafiori et al. 2024) and *Qwen2-72B-Instruct* (Yang et al. 2024), using fewer average active parameters (approximately 30-50 compared to 70).

However, domain-based routing has significant limitations when queries do not fit into predefined categories. Performance often deteriorates substantially in "other" or general domains, dropping from 77–97% accuracy in specialised areas to as low as 53% for "other" queries (Simonds et al. 2024). This limitation highlights the importance of a thorough domain definition and the difficulty of dealing with edge cases in categorical approaches.

Focusing specifically on the knowledge domain rather than the individual complexity of queries improves the router's ability to generalise on out-of-distribution datasets (Y. Wang et al. 2024). This approach allows less restrictive routing rules by linking a routing candidate to more general concepts (i.e., the knowledge domain), resulting in a 3.6% increase in accuracy compared to the direct mapping of queries to experts.

4.2.3 Query Complexity Inference. Moving beyond domain classification, complexity-based routing recognises that the intrinsic difficulty of individual queries often matters more than the topical domain for routing decisions, and attempts to capture this. By categorising user queries into complexity levels, the routing process becomes a classification task (Ding et al. 2024; Malekpour et al. 2024; Srivatsa et al. 2024; Stripelis et al. 2024; C. Wang et al. 2024). Query-level routing significantly outperforms domain-based approaches on related datasets, achieving 64.3% compared to 52.2% accuracy, and even surpassing much larger individual models (C. Wang et al. 2024). Remarkably, despite using a set of small LLMs (under 9B parameters), it surpasses the performance of a much larger LLM, *Llama-3-70b* (Grattafiori et al. 2024). However, this improvement comes at the cost of generalisation ability, with reduced performance on out-of-distribution datasets.

The definition of query complexity represents a critical design choice that fundamentally shapes system performance. The first approach measures complexity through performance differences between models of varying sizes (Ding et al. 2024) by computing the *BART* score (W. Yuan et al. 2021). This score calculates the probability of generating a sequence given a reference sequence with a *seq2seq* model, called *BART* (M. Lewis et al. 2020). When parameter size differences are substantial, such as between *FLAN-T5* (800m) (Chung et al.

2024) and *LLaMA-2-13B* (Touvron et al. 2023), 40% of queries were successfully routed to the smaller model with only 10% quality degradation. More remarkably, when differences are subtle, such as between *LLaMA-2-13B* and *GPT-3.5-turbo*, 20% of queries could still be handled by the smaller model with less than 1% quality reduction.
Alternative complexity definitions focus on previous successes rather than sequence probability (Malekpour et al. 2024; Stripelis et al. 2024). Malekpour et al. (2024) demonstrated that classifying the cheapest LLM capable of handling a query could reduce costs by 1.4 times, albeit with a loss of nearly six performance points compared to *GPT-4o-mini* (55.2% vs 61.0%). They showed that this strategy resulted in lower successful SQL generation than the similarity retrieval method discussed previously. However, it is important to note that this classification strategy relied more on smaller LLMs, which performed poorly in generating SQL queries. A better selection of LLM might yield different results (i.e., SQL Expert LLM). Stripelis et al. (2024) reduces costs by 30% and latency by 40% compared to stand-alone generalist LLM (*Fox-1.6B* (Z. Hu et al. 2025)) or experts LLM across various domains. The robustness-based approach to complexity assessment proposed by Srivatsa et al. (2024) evaluates LLM adequacy by generating multiple responses and comparing them to reference answers.[8] An LLM is considered adequate for a query if the majority of its responses align with ground truth. Although the router exhibited reduced latency, it did not achieve the same accuracy as the best-performing LLM, *gemma-7b*, suggesting that robustness may not fully capture query complexity nuances.

Complexity inference is treated as an optimisation problem in more sophisticated approaches. X. Wang et al. (2025) proposed *MixLLM*, a method to infer a cost-quality score and route to the routing options that maximise this score. It infers cost-quality scores using fine-tuned embeddings that minimise distance between sentences from the same knowledge domain. Then, they infer the output length for each query and each LLM and calculate the expected cost. Their weighted linear combination of quality, cost, and uncertainty scores, adjusted for user budget constraints, achieved 97% of GPT-4 accuracy at only 24% of the associated cost. It outperformed several methods proposed in the survey, including *RouterBench* (Q. J. Hu et al. 2024), *Zooter multi-perceptron* (Lu et al. 2024), Sakota et al. (2024)'s supervised strategy, BERT-based multi-classifier (Ong et al. 2025), *MetaLLM* (Nguyen et al. 2024), *OptLLM* (Y. Liu et al. 2024), and *AutoMix* (Aggarwal et al. 2024).

Direct performance score inference represents another approach, treating routing as a regression rather than classification task (Q. J. Hu et al. 2024). Somerstep et al. (2025) proposed to infer the expected cost and quality of LLM for specific queries using either a *KNN* or a fine-tuned *RoBERTa* model. Their strategy achieved similar accuracy (95%) to *GPT-4* on RouterBench (Q. J. Hu et al. 2024) but at 20% of its cost. Both the *KNN* and fine-tuned *RoBERTa* (L. Zhuang et al. 2021) models showed comparable performance, each outperforming Ong's preference-based matrix factorisation and fine-tuned *RoBERTa* multi-classifier (Ong et al. 2025). Q. J. Hu et al. (2024) derived performance scores as $Performance_{i,j} = \lambda \cdot P_{i,j} - cost_j$, where $P_{i,j}$ represents the predicted ability of model M_j to answer query q_i, λ represents user willingness to pay, and $cost_j$ denotes the model's cost [9]. This regression-based routing achieved performance comparable to *GPT-4* or *Claude V2* on standard datasets while significantly reducing costs. Sakota et al. (2024) extended this concept by proposing three selection strategies: maximising performance regardless of cost, selecting models that exceed performance thresholds while minimising expense, or optimising within budget constraints through integer linear programming. Maximising the performance score without considering cost achieves an accuracy equivalent to the best-performing model, *text-davinci-2*. However, this approach incurs a cost comparable to routing to the highest-performing model, resulting in an 11% cost reduction. By implementing either the threshold approach or the cost-sensitive method, they achieved comparable accuracy at a significantly reduced cost (approximately 62% decrease). Y. Liu et al. (2024) introduced *OptLLM*, a routing strategy that first infers the expected accuracy of each candidate LLM for a given query. Then, it constructs a

[8] kvadityasrivatsa/llm-routing
[9] withmartian/routerbench

set of Pareto optimal solutions using heuristic-based multi-objective optimisation, simultaneously maximising performance and minimising cost. During inference, the query can be routed to the Pareto-optimal solution that meets the user specific requirements, such as budget constraints. Across different datasets, *OptLLM* achieves accuracy comparable to the best single LLM while significantly reducing computational cost.
The challenge of generalising to tasks that have not been seen before has led to the development of approaches for out-of-distribution routing. Shnitzer et al. (2024) implemented a collection of binary classifiers to determine whether a model M_i within a set of n LLMs $\mathcal{M} = \{M_1, ..., M_n\}$ could provide an answer that matches the reference answer for a given query. However, the methodology for labelling the training set is not explicitly described. This model estimates the probability that a binary classifier's prediction is correct for a specific data point within a given task, reflecting the model's uncertainty. They demonstrated that these strategies outperformed the best model on average, *Llama-2-70B* (Touvron et al. 2023), by enabling the selection of smaller models that could provide adequate answers.

The architecture used for supervised routing varies significantly. When explicitly detailed, most studies employ *BERT*-family models as backbones, adapting them for supervised training: *BERT-based* (Devlin et al. 2019; Ong et al. 2025; Stripelis et al. 2024), *RoBERTa* (Ning et al. 2024; Srivatsa et al. 2024; L. Zhuang et al. 2021), *DistilBERT* (Malekpour et al. 2024; Sakota et al. 2024; Sanh et al. 2020; Srivatsa et al. 2024). The choice of backbone model highly influence the effectiveness of these classification architectures: Feng et al. (2025) demonstrated that replacing the *DeBERTa* model (He et al. 2023b) with the larger *RoBERTa* architecture (L. Zhuang et al. 2021) enhanced the performance of the *HybridLLM* framework. Alternative implementations have employed *T5* (Jeong et al. 2024; Raffel et al. 2020; Srivatsa et al. 2024), *multi-layers perceptron* (Q. J. Hu et al. 2024; S. Lee et al. 2025; Somerstep et al. 2025; Stripelis et al. 2024; T. Zhang et al. 2025), *Random Forest* (Srivatsa et al. 2024), *optimisation programmes* (Dekoninck et al. 2025; Y. Liu et al. 2024; X. Wang et al. 2025), or *K-NN* (Q. J. Hu et al. 2024; Shnitzer et al. 2024; Somerstep et al. 2025; Stripelis et al. 2024; T. Zhang et al. 2025). One notable difference from other *BERT*-based approaches is Mohammadshahi et al. (2024)'s *Routoo* Orchestrator[10], which employs decoder-only LLMs for query encoding rather than bidirectional language models (i.e., *BERT* models (Devlin et al. 2019)). This approach leverages the autoregressive nature of these models by extracting representations from predefined last tokens (typically '<\s>' or '<EOS>'). These embedding strategies have demonstrated high performance, with top-ranked models on the MTEB leaderboard[11] being decoder-only embedding models, such as Nvidia's *NV-Embed-v2* (C. Lee et al. 2025) and BAAI's *bge-en-icl* (C. Li et al. 2025). However, these require significant computing resources due to their larger size. Decoder-only models are typically fine-tuned for dense retrieval (C. Lee et al. 2025; C. Li et al. 2025; Z. Liu et al. 2024; Ma et al. 2024; Muennighoff et al. 2025; L. Wang et al. 2024). The researchers improve model selection by identifying a subset of LLMs that perform best across a given set of queries, aiming to select models whose strengths complement each other for more effective routing. The *Routoo* approach demonstrated superior accuracy on MMLU (Hendrycks et al. 2021) compared to larger standalone LLMs like *Llama2-70B* (75.9% vs 69.9%) and *Mixtral-8x7B* (75.9% vs 70.6%) at similar or lower costs. Although this strategy employs lightweight methods for query categorisation, it requires significant and unnecessary resources since it necessitates hosting and running a large decoder-only model for encoding.

Supervised routing success depends on complementary capabilities among available LLMs (Srivatsa et al. 2024). This limitation is discussed in the Section 5.3.3. Yue et al. (2024) found that a *RoBERTa* model fine-tuned to route between *GPT-3.5* and *GPT-4* was less effective than prompt-based routing, indicating that routing candidate selection may influence supervised routing effectiveness. The supervised routing paradigm extends beyond simple

[10] Leeroo-AI/leeroo_orchestrator
[11] MTEB leaderboard

generation tasks to encompass guardrails (S. Lee et al. 2025), retrieval strategies (Jeong et al. 2024), and prompting techniques (Ning et al. 2024). S. Lee et al. (2025) implemented *SafeRoute*, dynamically switching between guardrail models based on complexity, achieving the larger model's F1 score while significantly reducing latency. Jeong et al. (2024) demonstrated similar results with their *Adaptive-RAG*[12] framework, routing between no-retrieval, single-step naive RAG, and multi-step RAG methods based on complexity assessment. Similarly, in Ning et al. (2024)'s study, the model determines whether a query can be answered using complex prompting approach, treating it as a binary classification task[13]. The authors found that incorporating a router between prompts is more effective than applying their complex prompting to all queries. However, no comparison was made between the efficacy of their approach and that of a basic prompt as a baseline. Therefore, it is difficult to ascertain the relative efficacy of this approach.

4.2.4 Knowledge Graphs. The main drawback of most of the routing methods discussed is that they cannot be generalised to new routing options. As noted by Feng et al. (2025), these methods rely on a transductive learning framework, which involves learning specific rules from a corpus and applying them to particular cases, similar to those encountered during training. In rapidly evolving environments where new options frequently emerge, maintaining such architectures becomes prohibitively costly, necessitating frequent retraining whenever new models are introduced. To address this generalisation challenge, Feng et al. (2025) proposed an inductive graph framework, *GraphRouter*. This framework focuses on learning contextual information regarding task interactions and models, ultimately enhancing generalisation. The graph consists of three types of nodes: task, query, and LLM. To initialise task and LLM nodes, they first generate their descriptions and include additional calling cost information for the LLM nodes. Then, they encode these descriptions using a *BERT*-like model (Devlin et al. 2019). Query nodes are also encoded using the same model. Edge features capture relationships between nodes. Task-query edges indicate the relevance of queries to specific tasks (e.g. maths-related queries in GSM8K contexts), while LLM-query edges provide performance scores adjusted for desired costs by combining cost and performance metrics. The system implements a two-layer graph attention network with 32-dimensional hidden layers to update node representations using local neighborhood information. Subsequently, selecting an LLM can be reframed as an edge prediction between the query and LLM nodes.
They demonstrated a performance surpassing largest standalone LLM on a multi-task dataset while requiring lower costs. Their results also exceeded those of prompt-based routing, *HybridLLM* (Ding et al. 2024), *FrugalGPT* (L. Chen et al. 2023), and a bandit-based model. Notably, with new LLM options, they achieved a performance improvement of 4 points over *FrugalGPT* and 21 points over *HybridLLM*.

4.2.5 Answer Confidence Inference. In this strategy, the likelihood that the generated answer is correct is estimated by the router at each iteration, with routing decisions based on confidence thresholds rather than query characteristics. L. Chen et al. (2023) proposed *FrugalGPT*[14], which infers the probability that the generated answer is correct using a *DistilBERT* regression model (Sanh et al. 2020). The authors reported that their framework could save between 59% and 98% of costs while maintaining similar accuracy to larger standalone models, such as *GPT-4*. This strategy is often used for comparisons but underperforms against alternatives, such as LLM-based repeated calls routing (Aggarwal et al. 2024) and graph-based supervised routing (Feng et al. 2025). Moreover, even lower-resource LLM-based strategies, such as the token probability method (Ramírez et al. 2024), match its efficacy.

[12] starsuzi/Adaptive-RAG
[13] imagination-research/sot
[14] stanford-futuredata/Frugalgpt

4.3 Reinforcement Learning-based Routing

Supervised or similarity-based may not be able to adapt to new routing options or a new context, such as changes in the way users express themselves. This limitation arises because the routing process needs to learn from interactions (Sutton and Barto 2014). Reinforcement learning (RL) addresses this issue by treating routing as a sequential decision-making problem, whereby the router learns to select the optimal model through environmental feedback (Sutton and Barto 2014). In this paradigm, routing is framed as the selection of actions (models) based on the current state (user queries), with the aim of maximising rewards while respecting cost constraints.

4.3.1 Stateless Algorithms. Stateless algorithms offer a low-resource alternative by directly mapping rewards to actions. The Stochastic Learning Automaton (SLA) of the *PickLLM* framework demonstrates how this strategy works by maintaining probability distributions over candidate models and updating them based on interaction outcomes (Sikeridis et al. 2024) using a basic linear reward-inaction scheme (Narendra and Thathachar 1974): increasing the probability of selecting the LLM with the highest reward while decreasing the probability of other candidates. This strategy significantly reduces cost and latency compared to using the most expensive option or randomly selecting a model. Specifically, the SLA option reduces latency more consistently than the Q-learning option. Regarding accuracy, the model performed slightly better than *Llama-2-70B* and *Mixtral-8x7B* (A. Q. Jiang et al. 2024) on a medical subset and a computer science subset. However, especially when compared to *Mixtral-8x7B*, it underperformed on a subset of Reddit. This discrepancy may be due to the absence of *Mixtral-8x7B* in the provided set of options.

Instead of relying on action probabilities, one could optimise selection using expected rewards, referred to as *Q-values* (Kaelbling et al. 1996). In Q-learning, an agent iteratively updates its *Q-values* based on the rewards received by interacting with the environment (Kaelbling et al. 1996). The agent selects the LLM candidate with the highest expected reward, though it suffers from exploitation bias where successful actions dominate future selections. Epsilon-greedy exploration mitigates this by balancing known good choices with random exploration (Sikeridis et al. 2024). This technique selects a random action with a probability of ϵ and chooses the action with the highest expected reward, as indicated by *Q-values*, with a probability of $(1 - \epsilon)$ (Sikeridis et al. 2024). This strategy effectively reduces both cost and latency compared to purely random selection or routing only to standalone LLMs. Performance remains comparable to SLA approach in terms of accuracy.

Stateless strategies effectively converge on the LLM candidate that best fits the data, resource requirements, or context at a given time t (Sikeridis et al. 2024). However, they are poorly suited for real-world production environments where context evolves over time. These methods cannot adapt their actions to specific contextual changes

4.3.2 State-based Algorithms. State-based approaches are a significant conceptual advance because they integrate contextual information into routing decisions. Rather than learning a single optimal model, these systems determine which models perform best in specific situations.
Nguyen et al. (2024) proposed the *Meta-LLM* framework, which incorporates a contextual multi-armed bandit (MAB) model with a discrete set of actions, or arms, that can be selected at each step. Each arm represents a candidate LLM that the router may choose. Since the reward associated with each arm is initially unknown, the model learns over time by selecting an arm, observing the resulting rewards, and updating its expected reward. The goal is to learn to select the optimal arm in order to maximise cumulative reward over time. The MAB model achieved comparable accuracy to the most expensive and high-performing LLM, *text-davinci-002* (91.0 versus 90.9), while incurring significantly lower costs (0.3 versus 4.8 $ per 10,000 queries). The authors explain that some queries were correctly classified by small LLMs but not by large ones. In another trials, MAB matched the performance of the most performant LLM, *Llama-2-7B* (91.6 versus 91.5), and even outperformed the most

expensive model, *Claude Instant* (91.6 versus 87.6), while requiring drastically lower costs: *MetaLLM* at 0.5 $ per 10,000 queries, *Claude Instant* at 2.5 $ per 10,000 queries, and *Llama-2-7B* at 2.4 $ per 10,000 queries.

Y. Li (2025) extended contextual bandits by creating model identity vectors that encode performance patterns across standardised examples, in a manner similar to the approach adopted by Jitkrittum et al. (2025). The primary goal is to learn a performance representation based on a consistent corpus to obtain future candidate representations by running them on the same sample. The algorithm incorporates user preferences through proximal policy optimisation (Schulman et al. 2017). When comparing this strategy with an implementation of *RouteLLM* (Ong et al. 2025), their approach demonstrated superior performance. Notably, the *RouteLLM* implementation did not outperform random routing. Future research should compare the complexity of this reinforcement learning-based architecture with the simplicity of the Jitkrittum et al. (2025) approach to determine whether such sophisticated reinforcement learning techniques are necessary to achieve dynamic routing.

4.3.3 Reward-based Inference. Hari and Thomson (2023) noted that creating a Q-table to map the actual reward r_{M_i} for selecting an LLM M_i over all queries would be impractical. Instead of learning through direct interaction, these methods use separate reward models to predict which LLM will perform best for a given query. In their *Tryage* framework, Hari and Thomson (2023) infer expert model losses (rewards) by training a *BERT-small* model (Bhargava et al. 2021; Turc et al. 2019) on a dataset containing queries and corresponding expert model losses. This strategy was taken a step further by Lu et al. (2024) with *Zooter*, a regression model trained from *mdberta-v3-base* (He et al. 2023b) through knowledge distillation from a reward model, *QwenRM* (Bai et al. 2023). Using a reward model proves to be an effective approach to the routing problem. Hari and Thomson (2023) discovered that their framework achieved greater routing effectiveness to the "ideal" expert model (50.8% accuracy) compared to a stand-alone *GPT-3.5* (23.6%) or the fine-tuned LLM *Gorilla* (Patil et al. 2024) (10.8%). There are no explicit details on the selection criteria for the ideal expert model.

Beyond demonstrating technical feasibility, Lu et al. (2024) revealed two critical insights about the design of effective routing systems. Firstly, the complementarity of models matters. The scores of the independent models vary significantly depending on the dataset, with some models excelling on specific datasets; for instance, *WizardLM* performs well on the Flask Dataset (Ye et al. 2024), while *Llama-2-Chat* (Touvron et al. 2023) performs on AlpacaEval. Nevertheless, Zooter achieves an average performance comparable to the best model across each dataset. Secondly, diversity in parameter size becomes crucial for complex tasks, as evidenced by performance gaps when all candidate models were of a similar size. In a benchmark comprising MMLU, GSM8K, and HumanEval, all open-source models exhibited poor performance, which constrained *Zooter*'s overall effectiveness. However, *Zooter*'s performance closely aligns with that of *GPT-4*, apart from the last benchmark, where *GPT-4* significantly outperforms it (32.3 for *Zooter* versus 88.3 for *GPT-4*). Changing the reward model ranking did not improve *Zooter* performance on this particular benchmark. This difference may be related to the tasks' complexity, which may be too challenging for 13B parameter models, given that *GPT-4* is much larger than the LLMs integrated into the router.

4.4 Generative-based Routing

This category of strategy uses the emerging capabilities of LLMs to make routing decisions, taking advantage of their ability to perform unsupervised multitasking and generalise to new situations (Radford et al. 2019; J. Wei et al. 2022).

4.4.1 Token Probability. The most resource-efficient approach in this category involves measuring output uncertainty at the token level (Ramírez et al. 2024). Actually, it achieves this in a low-resource manner by measuring *output uncertainty* or *margin*. Researchers calculate the margin between the probabilities of the first and second most likely tokens using the list of possible tokens returned by an LLM for the token at the first position

of the generation. This method represents a significant improvement in resource requirements over sequence-level approaches like the *BART* score, as it requires only partial generation rather than complete output generation. A larger LLM is called if the margin exceeds a threshold established by a budget-based criterion. This approach has proven to be effective in maximising performance while minimising costs while routing between smaller and larger LLM, including regression models, *HybridLLM* (Ding et al. 2024) and *FrugalGPT* (L. Chen et al. 2023). The differences are most significant when using the small-large LLM pair *GPT-3* and *GPT-4*. This suggests that the token probability approach is most valuable when there is a significant capability gap between models, allowing clear signals of uncertainty to emerge. *Frugal-GPT* performs better on several datasets of classification tasks (e.g. ISEAR, RT-Pol) with smaller LLM pairs (*Mistral-7B-Instruct-v0.2 & Mixtral-8x7B-Instruct-v0.1*, *Llama-2-13B*-hf & *Llama-2-70b-hf*) but not on Q/A and reasoning tasks.

4.4.2 Sequence Probability. Sequence-level probability assessment represents a natural extension of token-level approaches, considering the cumulative uncertainty across entire generated sequences. C.-H. Lee et al. (2024) proposed using the normalised sequence-level probability of a smaller LLM when determining whether to route a query to a larger LLM. However, their findings reveal a critical limitation of this approach: it tends to rely too heavily on larger models, essentially negating the efficiency benefits that routing is designed to provide. Consequently, this approach is less effective in reducing call frequency to the larger LLM than the supervised methods discussed earlier.

4.4.3 Prompt-based Routing. Prompt-based routing is the most accessible form of generative routing. It requires minimal infrastructure and leverages the natural language understanding capabilities of LLMs. This approach operates through two distinct paradigms: function-calling for direct routing decisions and verbalised confidence assessment for uncertainty-aware routing.

Function-calling routing provides LLMs with routing candidates and task descriptions, allowing them to decide which model to route to based on the characteristics of the query. Routing is performed based on the candidate returned by the LLM (Ning et al. 2024; Shen et al. 2023). Although it leverages an LLM, it is more energy-efficient and requires fewer resources than fine-tuning an LLM. Shen et al. (2023) proposed *HuggingGPT*[15], a framework for task planning and execution (Shen et al. 2023). An LLM is prompted to select between different models based on their descriptions. This approach offers the advantage of performing inference with minimal examples in the prompt, known as "few-shot inference", or without any examples at all, known as "zero-shot inference". This is particularly useful when limited resources are available for annotation. Studies by Ning et al. (2024) demonstrate that even highly performant LLM, such as *GPT-4* did not outperform smaller, task-specific models like fine-tuned *RoBERTa*. This limitation stems from the fact that it is difficult to make accurate routing decisions without observing the model's actual performance in response to a specific query.

Verbalised confidence routing addresses this limitation by allowing models to express uncertainty after attempting to answer queries (Z. Li et al. 2024; M. Xiong et al. 2024). Z. Li et al. (2024) proposed *Self-Route*. They instructed the model to indicate whether additional context, retrieved using a RAG strategy, is sufficient. Otherwise, they suggest using the entire document from which the chunks were extracted. Their findings reveal that implementing this strategy outperformed a naive RAG system in terms of accuracy for the most commonly used LLMs (i.e. *GPT-4o* and *GPT-3.5-turbo*). The strategy proposed also outperformed sending large contexts for *GPT-3.5*, which has a smaller context window (16k). In contrast, performance between *Self-Route* and sending a large context was comparable for *GPT-4o*, which has a larger context window (128k) while using an average of 61% fewer tokens. Finally, for *gemini-1.5-pro*, which features a context window of 1M tokens, sending only large contexts appears to perform better than *Self-Route*, although the latter's performance remains reasonably close. The results for *gemini-1.5-pro* stem from its large context window, which allows to include large contexts without

[15] microsoft/JARVIS

truncation. While this method shows promise, it suffers from a critical weakness: LLMs tend to be overconfident in their certainty assessments (M. Xiong et al. 2024). This overconfidence can lead to suboptimal routing decisions, with some studies showing that verbalized confidence performs no better than random routing (Chuang et al. 2024).

4.4.4 LLM Fine-Tuning. When computational resources are available, fine-tuning LLMs specifically for routing tasks offers the potential for superior performance through task-specific optimization. This approach encompasses several strategies: domain classification(J. Liu et al. 2024), complexity scoring (Ong et al. 2025), and API call generation (Patil et al. 2024).

The domain classification approach demonstrates remarkable potential, with J. Liu et al. (2024) achieving accuracy improvements from 15% to nearly 100% on MMLU through supervised fine-tuning of a *Qwen1.5-1.8B-Chat* model (Bai et al. 2023).[16] This meta-model categorises the prompt, and the corresponding pre-trained expert associated with that category then generates a response. However, this approach requires retraining when new domains are introduced, limiting its adaptability. In contrast, Ong et al. (2025) fine-tuned a *Llama-3-8B* (Grattafiori et al. 2024) on a scoring task to evaluate both the complexity of a query and the model's ability to answer it through LLM evaluation (Ong et al. 2025) and found that fine-tuning for complexity scoring provided only a small improvement on non-LLM techniques, suggesting that the benefits of fine-tuning depend heavily on the specific task. The authors' differing levels of optimism regarding the fine-tuning of an LLM for classification or regression tasks may also stem from Ong et al. (2025) comparing this technique with other optimised approaches, while J. Liu et al. (2024) evaluated it against the same standalone LLM without fine-tuning. The API call generation approach by Patil et al. (2024) treated routing as a code generation problem.[17]. Their fine-tuned *Llama-7B* model achieved substantial improvements over larger models in zero-shot scenarios, demonstrating that task-specific fine-tuning can enable smaller models to outperform larger general-purpose models in specialized routing contexts. It outperforms larger models after fine-tuning, achieving an average improvement of 35 points over *GPT-3.5-turbo-0301* and 46 points over *GPT-4-0314*.

4.4.5 Repeated Calls. The repeated calls approach relies on model confidence, which is demonstrated by consistent outputs across multiple generation attempts. By generating multiple responses at elevated temperatures and analyzing their consistency, this method provides robust confidence estimates without requiring additional model training or complex probability calculations. Then, routing to the next model can be triggered when the LLM exhibits low confidence. Smaller LLMs tend to answer simple questions consistently but show inconsistencies when confronted with more complex questions (Yue et al. 2024).

Two approaches implemented this approach (Aggarwal et al. 2024; Yue et al. 2024). Aggarwal et al. (2024) introduced a three-step approach called *Automix*[18]. First, a smaller model is used to generate an answer to a query based on related context. Next, the same model is prompted with a few-shot meta-prompt, called the verification prompt, to verify whether the answer is consistent with the context. To estimate the model's alignment with the context, a confidence score is calculated by generating the answer multiple times at a high temperature and determining the proportion of consistent answers. Based on this confidence score, the router either retains the current answer or calls a larger model. This procedure is repeated until the confidence score reaches an acceptable level or all models in the series have been tested. The authors employed two routing strategies: a more complex strategy based on a partially observable Markov decision process (POMDP), and a simpler strategy based on a confidence-cost/quality trade-off threshold. The authors reported that *Automix* (Aggarwal et al. 2024) achieves a

[16] godcherry/ExpertTokenRouting
[17] ShishirPatil/gorilla
[18] automix-llm/automix

higher F1 score on the QASPER and COQA datasets and superior accuracy across a range of costs compared to *HybridLLM* (Ding et al. 2024), *FrugalGPT*, (L. Chen et al. 2023) or stand-alone models such as *GPT-4* and *Llama-2-13B* (Touvron et al. 2023), particularly the routing based on the POMDP approach. Even in low-resource scenarios with a small training dataset size, their method significantly outperforms both *HybridLLM* (Ding et al. 2024) and *FrugalGPT* (L. Chen et al. 2023). However, these results require cautious interpretation, as X. Wang et al. (2025) found that the method did not outperform random routing on RouterBench (Q. J. Hu et al. 2024).
The *Mixture-of-Thoughts* approach (Yue et al. 2024) takes a different perspective, focusing on consistency across different reasoning representations rather than answer correctness [19]. They assessed response consistency across different reasoning-based prompting.
In both studies, users have to call an LLM multiple times for each query, and despite their efficiency, these approaches prove to be resource-intensive.

4.4.6 Code Execution. When it comes to code generation tasks, successful code execution provides an unambiguous confidence signal, eliminating the ambiguities inherent in other assessment methods. *EcoAssistant*[20], an iterative multi-agent code generator designed to query external knowledge for question and answering, demonstrates this principle by using execution success as a routing criterion, forwarding failed code generation attempts from smaller to larger models J. Zhang et al. (2023). The authors reported that their framework generates more successful code snippets at a lower cost than using *GPT-4*. Although more expensive than using *GPT-3.5-turbo*, *EcoAssistant* significantly exceeded the percentage of successful code generation.

5 Discussion

In this work, we described a wide range of routing strategies, from learning similarities to fine-tuning LLM. Most of the strategies presented in this paper follow the pre-generation approach, employing frugal mechanisms such as similarity learning or supervised learning. In contrast, post-generation strategies are generally more resource-intensive, as they often involve generating multiple responses for a single query.

This survey demonstrates that the routing problem can be effectively addressed through low-resource solutions that maintain generalisation capabilities without incurring significant financial or computational costs (Feng et al. 2025).

5.1 Industrial Consideration

In addition to providing a summary of the current state of scientific contributions, it is essential to also consider the industrial landscape. In other words, *what current strategies are companies implementing and disseminating in light of the findings from this review?* It seems that most companies aim to direct users queries to specific tools, prompts or scripts. These routing methods are generally based on a rather simplistic methodology, including conditional approach[21], similarity routing[22], or simply a prompt-based classification system[23] Some companies leverage more sophisticated architecture by employing an LLM to generate synthetic data used to implement a classification-based routing with a small classifier[24]. Many of the mainstream LLM providers propose prompt-based routing, such as AWS[25] or OpenAI[26]. This survey aims to improve the transfer of various lightweight routing strategies identified from research to industry.

[19] MurongYue/LLM_MoT_cascade
[20] JieyuZ2/EcoAssistant
[21] Haystack's ConditionalRouter, Haystack's FileTypeRouter
[22] aurelio-labs/semantic-router
[23] LangChain's router, Llamaindex's router
[24] lamini-ai/llm-routing-agent
[25] awslabs/multi-agent-orchestrator
[26] openai/swarm

5.2 Key Challenges

Although routing systems are an efficient strategy to optimize resource consumption in LLM-based systems, their implementation faces several critical challenges that must be addressed. In this section, we examine five fundamental challenges: the paradoxical resource consumption of the router infrastructure itself, the importance of generalising to new models, the annotation bottleneck to train routers, the complexity of managing evolving candidate pools, and the vulnerability to adversarial attacks.

5.2.1 Routing Resource Savings Versus Technical Debt. Routers are designed to maximise performance while minimising resource consumption. However, as demonstrated earlier, many effective routing strategies themselves demand significant resources like fine-tuning and deploying LLM-based routers (J. Liu et al. 2024; Ong et al. 2025; Patil et al. 2024). This creates a paradox: routing infrastructure designed to reduce costs may ultimately consume more resources than it saves in the long term. This trade-off is a clear example of technical debt, where short-term performance gains come at the expense of increased long-term resource requirements and maintenance costs (Avgeriou et al. 2016). Consequently, before implementing a specific routing strategy, experts must carefully evaluate whether the operational savings achieved justify the computational and engineering costs of building, training and maintaining the router.

5.2.2 Generalisation to New Candidate Models. Traditional routing systems, including most strategies examined in this work, are inherently static, a limitation that limits their ability to generalise to new candidate models. Introducing new routing candidates, whether models, prompts, workflows, or modules, typically requires retraining the entire routing infrastructure. This limitation poses one of the most significant challenges for practical applications. Given the rapid pace of improvement in LLM capabilities (Xiao et al. 2025), routing systems that cannot efficiently adapt to new candidates quickly become outdated.
Recent research suggests promising alternatives that support the addition of new components without the need for retraining. Some strategies includes projecting routing candidates into a shared space, which allows for generalisation to LLM candidates that have not been seen before (Feng et al. 2025; Jitkrittum et al. 2025; Y. Li 2025).

5.2.3 The Routing Data Annotation Bottleneck. Training routing systems presents a fundamental scaling challenge: the cost of annotation increases with both the number of queries and the number of candidates. Consider the EmbedLLM benchmark (R. Zhuang et al. 2025), which evaluates 112 LLM candidates on the samples from MMLU(Hendrycks et al. 2021) - approximately 16,000. Building a routing dataset for this configuration requires generating responses from each candidate for every query - approximately 1.8 million generations. These responses must then be evaluated, either through human annotation or LLM-as-a-Judge approaches, with the latter doubling the generation cost. The annotation requirements for this far exceed those for traditional supervised learning, where each query requires only a single response label. More critically, the annotation cost accumulates over time: adding a new candidate to an existing pool requires generating and evaluating responses across the entire query set, while adding new queries necessitates evaluation across all candidates. Even similarity-based routing strategies cannot escape this bottleneck. While they avoid training a routing model directly, they still require annotated query-candidate pairs for their retrieval corpus. Each new candidate requires evaluation across existing queries, and each new query must be evaluated across all candidates to maintain an effective corpus.
This annotation effort represents a significant practical barrier to deploying and maintaining routing systems in production environments.

5.2.4 Candidate Pool Management and Candidate Deprecation. Dynamic incorporation of new candidates is essential for keeping routers efficient. However, this raises a critical inverse problem: without removal mechanisms, the candidate pool could grow indefinitely, forcing practitioners to maintain an overly large and costly collection

of LLMs that is not necessary. This problem manifests in two ways. Firstly, LLMs become outdated as newer, superior alternatives emerge. Secondly, candidate pools often contain redundant LLM that provide no additional value. J. Yuan et al. (2025) demonstrated this redundancy empirically: removing 30% of candidates with redundant performances from the EmbedLLM benchmark (R. Zhuang et al. 2025) had no impact on routing performance, yet maintaining these unnecessary candidates would impose substantial operational costs in production. Addressing candidate removal requires mechanisms to forget deprecated models while preserving routing ability. Machine unlearning (S. Liu et al. 2025; Q. Wang et al. 2025; Xu et al. 2023) offers one approach to selectively forget depreciated candidates, but introduces substantial complexity by requiring careful balance between forgetting and retention. In contrast, simpler alternatives, such as unsupervised frameworks (Jitkrittum et al. 2025), naturally adapt to candidate evolution without the requirements of selective forgetting, making them more practical for routing systems.

5.2.5 Adversarial Attacks to Routing Systems. Adversarial attacks represent a security concern in machine learning systems, where malicious actors exploit model vulnerabilities to induce unintended behaviours. These attacks can be broadly categorized into two classes: evasion attacks, which manipulate inputs during inference to alter model behaviour, and data poisoning attacks, which corrupt training data to compromise model integrity (L. Lin et al. 2025; Zhou et al. 2022). Routing systems are vulnerable to both attack vectors.

In LLM-based architectures, routers can be seen as the control plane, managing data flow throughout the system[27] (Shafran et al. 2025). This central role makes them an attractive target for attacks. Shafran et al. (2025) recently described a concerning threat: the *LLM control plane integrity attack* (Q. Lin et al. 2025; Shafran et al. 2025). Unlike conventional evasion attacks that aim to corrupt final model outputs (Akhtar et al. 2021), this attack targets the routing mechanism itself in order to change the system behaviour. Adversaries craft malicious inputs designed to consistently trigger selection of the most expensive model, thereby inflating operational costs without necessarily affecting response quality. By compromising the control plane, attackers can force data flow patterns and resource allocation throughout the system, even when the underlying LLM produce correct outputs.

Beyond input manipulation, routers are also vulnerable to data poisoning attacks. Adversaries can poison public routing benchmarks to embed backdoors within the router and activate them using specific triggers (Q. Lin et al. 2025). Additionally, dynamic data poisoning attacks can compromise preference data to favour certain LLM selection behaviours for specific topics when customer feedback is continuously incorporated into the routing system (P. Zhao et al. 2025).

5.3 Future Directions

Several research topics in routing in LLM-based systems remain under-explored. We identify four areas for investigation: expanding routing objectives to include computational and environmental metrics beyond financial costs, establishing standardised evaluation frameworks for comparing routing strategies, designing candidate pools with complementary capabilities, and extending routing decisions to all system components.

5.3.1 Going Beyond Financial Costs. Most studies focus on optimising the trade-off between financial costs and answer quality (L. Chen et al. 2023) while overlooking other significant expenses, such as computational and ecological costs. Computational requirements and environmental impacts are intricately linked. Models with larger parameter numbers generally require more floating-point operations, resulting in increased energy consumption (Desislavov et al. 2023; Kaack et al. 2022). It is essential to minimise not only the financial costs but also the computational requirements, given the urgent need to mitigate the impact of LLMs on climate change (Kaack et al. 2022; Luccioni et al. 2024). This can be accomplished by implementing routing modules that activate more resource-intensive components only when necessary. Future work should consider incorporating

[27] IBM Control Plane Definition

the computational and environmental costs discussed in section 2.2 into the cost function C_M from the Equation 1.

5.3.2 Standardisation of Routing Strategy Experiments. The field lacks a consensus framework to evaluate routing strategies. Many proposals are only benchmarked against non-routing or self-defined baselines. This makes it difficult to assess or compare their effectiveness objectively. More recently, datasets have been introduced to allow the comparison of different routing architectures under the same conditions: RouterBench (Q. J. Hu et al. 2024), MixInstruct (D. Jiang et al. 2023), EmbedLLM (R. Zhuang et al. 2025), SPROUT (Somerstep et al. 2025). Future research should incorporate standardised comparison baselines: *(i) random routing, (ii) the most cost-effective LLM with the highest performance for each query (oracle routing), (iii)) the stand-alone best-performing LLM, and (iv) alternative routing strategies reviewed in this survey.* Such comparisons will determine whether the efficacy of routing derives from the available routing candidates or the routing architecture. The performance gap between the best stand-alone LLM and gold standard routing quantifies the theoretical margin for improvement. Direct comparison of stand-alone LLMs with routing systems will quantify the added value of routing architectures across benchmark datasets. Additionally, J. Yuan et al. (2025) identifies critical evaluation pitfalls in current routing benchmarks, including task redundancy, model capability redundancy (also discussed in the following subsection), and model dominance.

5.3.3 Using Complementary Routing Options. Routing to candidates with complementary rather than redundant skills is essential to optimise routing performance. For example, multiple models of the same size (e.g., 7B parameters) trained on a generalist corpus may exhibit some complementarity due to differences in their training datasets. However, we cannot expect significant performance enhancements for queries on specific or complex topics requiring advanced reasoning. The primary objective of routing is to maximise quality. Incorporating models with varying parameter sizes or trained on specialised domains allows the routing strategy to adapt effectively to various contexts. In addition, when evaluating different routing approaches, it is important to consider how complementary and efficient the different routing candidates are.

5.3.4 Consider All Steps in the LLM-based System as Routing Possibilities. Most studies focused on the generation step by selecting the most appropriate LLM to answer the user query. However, current LLM-based systems like conversational agents typically include several additional steps. The routing approach can be effectively applied during the embedding step in the RAG architecture. This includes routing between different embedding strategies, such as dense or sparse vectors, or the selection of fine-tuned embedding models (Y. Gao, Y. Xiong, X. Gao, et al. 2024). It also allows routing to databases tailored to specific topics, selecting appropriate similarity functions such as cosine similarity or *BM25*, and selecting the most appropriate prompting approaches (Y. Gao, Y. Xiong, X. Gao, et al. 2024). This framework can also be extended to facilitate routing between static knowledge sources, such as databases, and dynamic knowledge sources, such as website searches. Some authors have studied the use of routing for additional steps, such as context size selection (Z. Li et al. 2024), prompt strategies (Ning et al. 2024), and even alternative pipeline designs (Jeong et al. 2024). Viewing these systems as dynamic rather than traditional static models facilitates optimisation at each stage and enhances modularity (Y. Gao, Y. Xiong, M. Wang, et al. 2024). This transforms query answering into a singular dynamic process that depends on the specific query being addressed.

6 Conclusion

Routing in an LLM-based system can be defined as a process that aims to select components of the system that maximise performance while minimising cost. The definition of the cost to be minimised and the score function to be maximised are essential to build a routing algorithm. We classified routing strategies as pre-generation or post-generation. We highlighted that implementing a routing strategy to maintain performance while minimising

cost is efficient and does not necessarily require high resources. We highlighted the need to develop products based on the lightweight strategies discussed in this survey.

However, several critical challenges must be addressed for its implementation: the paradox of router infrastructure consuming more resources than it saves, the annotation bottleneck that scales with both queries and candidates, the difficulty of generalising to new models without retraining, managing evolving candidate pools while removing deprecated models, and vulnerability to adversarial attacks targeting routing decisions. The field needs standardized benchmarks with consistent baseline comparisons, cost functions that incorporate computational and environmental impacts beyond financial metrics, and routing systems designed with complementary rather than redundant candidates.

References

P. Aggarwal et al.. 2024. "AutoMix: Automatically Mixing Language Models." In: *The 38th Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=e6WrwIvgzX.

N. Akhtar, A. Mian, N. Kardan, and M. Shah. 2021. "Advances in Adversarial Attacks and Defenses in Computer Vision: A Survey." *IEEE Access*, 9, 155161–155196. doi:10.1109/ACCESS.2021.3127960.

P. Avgeriou, P. Kruchten, I. Ozkaya, and C. Seaman. Jan. 2016. "Managing Technical Debt in Software Engineering (Dagstuhl Seminar 16162)." *Dagstuhl Reports*, 6, (Jan. 2016). doi:10.4230/DagRep.6.4.110.

J. Bai et al.. 2023. *Qwen Technical Report*. (2023). arXiv: 2309.16609 (cs.CL).

A. P. Behera, J. P. Champati, R. Morabito, S. Tarkoma, and J. Gross. 2025. *Towards Efficient Multi-LLM Inference: Characterization and Analysis of LLM Routing and Hierarchical Techniques*. (2025). arXiv: 2506.06579 (cs.LG).

P. Bhargava, A. Drozd, and A. Rogers. 2021. "Generalization in NLI: Ways (Not) To Go Beyond Simple Heuristics." In: *Proceedings of the Second Workshop on Insights from Negative Results in NLP*. Ed. by J. Sedoc, A. Rogers, A. Rumshisky, and S. Tafreshi. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 125–135. doi:10.18653/v1/2021.insights-1.18.

C. Bouvard, M. Ciancone, A. Gourru, and M. Schaeffer. July 2024. "Derby LLM : Évaluation comparative des approches RAG et fine-tuning." In: *Actes de la 10 ème Conférence Nationale sur les Applications Pratiques de l'Intelligence Artificielle (APIA)* (Plate-Forme Intelligence Artificielle (PFIA)). Ed. by C. R. Ghislain Atemezing. AFIA-Association Française pour l'Intelligence Artificielle. AFIA-Association Française pour l'Intelligence Artificielle, La Rochelle, France, (July 2024), 38–47. https://hal.science/hal-04638460.

R. A. Bradley and M. E. Terry. 1952. "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons." *Biometrika*, 39, 324–345. doi:https://doi.org/10.2307/2334029.

W. Cai, J. Jiang, F. Wang, J. Tang, S. Kim, and J. Huang. 2025. "A Survey on Mixture of Experts in Large Language Models." *IEEE Transactions on Knowledge and Data Engineering*, 1–20. doi:10.1109/tkde.2025.3554028.

G. Caldarini, S. Jaf, and K. McGarry. 2022. "A Literature Survey of Recent Advances in Chatbots." *Information*, 13, 1. doi:10.3390/info13010041.

Z. Chai et al.. 2024. "An Expert is Worth One Token: Synergizing Multiple Expert LLMs as Generalist via Expert Token Routing." In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*. Ed. by L.-W. Ku, A. Martins, and V. Srikumar. Association for Computational Linguistics, Bangkok, Thailand, 11385–11396. doi:10.18653/v1/2024.acl-long.614.

Y. Chang et al.. 2024. "A Survey on Evaluation of Large Language Models." *ACM Trans. Intell. Syst. Technol.*, 15, 3, Article 39, 45 pages. doi:10.1145/3641289.

L. Chen, M. Zaharia, and J. Zou. 2023. *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. (2023). arXiv: 2305.05176 (cs.LG).

S. Chen, W. Jiang, B. Lin, J. Kwok, and Y. Zhang. 2024. "RouterDC: Query-Based Router by Dual Contrastive Learning for Assembling Large Language Models." In: *The 38th Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=7RQvjayHrM.

Y. Chen, J. Zhao, and H. Han. 2025. *A Survey on Collaborative Mechanisms Between Large and Small Language Models*. (2025). arXiv: 2505.07460 (cs.AI).

Z. Chen et al.. 2025. *Harnessing Multiple Large Language Models: A Survey on LLM Ensemble*. (2025). arXiv: 2502.18036 (cs.CL).

C.-H. Chiang and H.-y. Lee. 2023. "A Closer Look into Using Large Language Models for Automatic Evaluation." In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by H. Bouamor, J. Pino, and K. Bali. Association for Computational Linguistics, Singapore, 8928–8942. doi:10.18653/v1/2023.findings-emnlp.599.

Y.-N. Chuang, H. Zhou, P. K. Sarma, P. Gopalan, J. Boccio, S. Bolouki, and X. Hu. 2024. *Learning to Route with Confidence Tokens*. (2024). arXiv: 2410.13284 (cs.CL).

H. W. Chung et al.. 2024. "Scaling Instruction-Finetuned Language Models." *Journal of Machine Learning Research*, 25, 70, 1–53. http://jmlr.org/papers/v25/23-0870.html.

S. K. Dam, C. S. Hong, Y. Qiao, and C. Zhang. 2024. *A Complete Survey on LLM-based AI Chatbots*. (2024). arXiv: 2406.16937 (cs.CL).

J. Dekoninck, M. Baader, and M. Vechev. 2025. *A Unified Approach to Routing and Cascading for LLMs*. (2025). arXiv: 2410.10347 (cs.CL).

R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo. 2023. "Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning." *Sustainable Computing: Informatics and Systems*, 38, 100857. doi:https://doi.org/10.1016/j.suscom.2023.100857.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 4171–4186. doi:10.18653/v1/N19-1423.

D. Ding, A. Mallick, C. Wang, R. Sim, S. Mukherjee, V. Rühle, L. V. S. Lakshmanan, and A. H. Awadallah. 2024. "Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing." In: *The 12th International Conference on Learning Representations*. https://openreview.net/forum?id=02f3mUtqnM.

A. E. Elo. 1978. *Ratings of Chess Players Past and Present*. Arco Pub.

S. Es, J. James, L. Espinosa Anke, and S. Schockaert. 2024. "RAGAS: Automated Evaluation of Retrieval Augmented Generation." In: *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, 150–158. https://aclanthology.org/2024.eacl-demo.16.

W. Fedus, B. Zoph, and N. Shazeer. 2022. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity." *Journal of Machine Learning Research*, 23, 120, 1–39. http://jmlr.org/papers/v23/21-0998.html.

T. Feng, Y. Shen, and J. You. 2025. "GraphRouter: A Graph-based Router for LLM Selections." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=eU39PDsZtT.

J. Fürnkranz and E. Hüllermeier. 2012. *Preference Learning*. Ed. by C. Sammut and G. I. Webb. Springer US, 1–7. doi:10.1007/978-1-4899-7502-7_667-1.

Y. Gao, Y. Xiong, X. Gao, et al.. 2024. *Retrieval-Augmented Generation for Large Language Models: A Survey*. (2024). arXiv: 2312.10997 (cs.CL).

Y. Gao, Y. Xiong, M. Wang, and H. Wang. 2024. *Modular RAG: Transforming RAG Systems into LEGO-like Reconfigurable Frameworks*. (2024). arXiv: 2407.21059 (cs.CL).

A. Grattafiori et al.. 2024. *The Llama 3 Herd of Models*. (2024). arXiv: 2407.21783 (cs.AI).

N. Guha, M. F. Chen, T. Chow, I. S. Khare, and C. Re. 2024. "Smoothie: Label Free Language Model Routing." In: *The 38th Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=pPSWHsgqRp.

S. N. Hari and M. Thomson. 2023. *Tryage: Real-time, intelligent Routing of User Prompts to Large Language Models*. (2023). arXiv: 2308.11601 (cs.LG).

P. He, J. Gao, and W. Chen. 2023a. *DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing*. (2023). arXiv: 2111.09543 (cs.CL).

P. He, J. Gao, and W. Chen. 2023b. "DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing." In: *The 11th International Conference on Learning Representations*. https://openreview.net/forum?id=sE7-XhLxHA.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. 2021. "Measuring Massive Multitask Language Understanding." In: *International Conference on Learning Representations*. https://openreview.net/forum?id=d7KBjmI3GmQ.

J. Hu, Y. Wang, S. Zhang, K. Zhou, G. Chen, Y. Hu, B. Xiao, and M. Tan. 2024. *Dynamic Ensemble Reasoning for LLM Experts*. (2024). arXiv: 2412.07448 (cs.AI).

Q. J. Hu, J. Bieker, X. Li, N. Jiang, B. Keigwin, G. Ranganath, K. Keutzer, and S. K. Upadhyay. 2024. "RouterBench: A Benchmark for Multi-LLM Routing System." In: *Agentic Markets Workshop at ICML 2024*. https://openreview.net/forum?id=IVXmV8Uxwh.

Z. Hu et al.. 2025. *Fox-1 Technical Report*. (2025). arXiv: 2411.05281 (cs.CL).

C. Irugalbandara, A. Mahendra, R. Daynauth, T. Arachchige, J. Dantanarayana, K. Flautner, L. Tang, Y. Kang, and J. Mars. 2024. "Scaling Down to Scale Up: A Cost-Benefit Analysis of Replacing OpenAI's LLM with Open Source SLMs in Production." In: *2024 International Symposium on Performance Analysis of Systems and Software*, 280–291. doi:10.1109/ISPASS61541.2024.00034.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. 1991. "Adaptive Mixtures of Local Experts." *Neural Computation*, 3, 1, 79–87. doi:10.1162/neco.1991.3.1.79.

S. Jain et al.. 2024. *Composition of Experts: A Modular Compound AI System Leveraging Large Language Models*. (2024). arXiv: 2412.01868 (cs.LG).

J. Jang, S. Kim, S. Ye, D. Kim, L. Logeswaran, M. Lee, K. Lee, and M. Seo. 2023. "Exploring the benefits of training expert language models over instruction tuning." In: *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 14702–14729. https://dl.acm.org/doi/abs/10.5555/3618408.3619008.

S. Jeong, J. Baek, S. Cho, S. J. Hwang, and J. Park. 2024. "Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity." In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies. Ed. by K. Duh, H. Gomez, and S. Bethard. Mexico City, Mexico, 7036–7050. doi:10.18653/v1/2024.naacl-long.389.

A. Q. Jiang et al.. 2024. *Mixtral of Experts*. (2024). arXiv: 2401.04088 (cs.LG).

D. Jiang, X. Ren, and B. Y. Lin. 2023. "LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion." In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Association for Computational Linguistics, Toronto, Canada, 14165–14178. doi:10.18653/v1/2023.acl-long.792.

R. Jiang, K. Chen, X. Bai, Z. He, J. Li, M. Yang, T. Zhao, L. Nie, and M. Zhang. 2024. *A Survey on Human Preference Learning for Large Language Models*. (2024). arXiv: 2406.11191 (cs.CL).

W. Jitkrittum et al.. 2025. *Universal Model Routing for Efficient LLM Inference*. (2025). arXiv: 2502.08773 (cs.CL).

L. H. Kaack, P. L. Donti, E. Strubell, G. Kamiya, F. Creutzig, and D. Rolnick. 2022. "Aligning artificial intelligence with climate change mitigation." *Nature Climate Change*, 12, 6, 518–527. doi:10.1038/s41558-022-01377-7.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. 1996. "Reinforcement learning: a survey." *Journal of Artificial Intelligence Research*, 4, 1, 237–285.

E. Kamalloo, N. Dziri, C. Clarke, and D. Rafiei. July 2023. "Evaluating Open-Domain Question Answering in the Era of Large Language Models." In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by A. Rogers, J. Boyd-Graber, and N. Okazaki. Association for Computational Linguistics, Toronto, Canada, (July 2023), 5591–5606. doi:10.18653/v1/2023.acl-long.307.

P. Le-Khac, G. Healy, and A. Smeaton. Jan. 2020. "Contrastive Representation Learning: A Framework and Review." *IEEE Access*, 8, (Jan. 2020), 193907–193934. doi:10.1109/ACCESS.2020.3031549.

C. Lee, R. Roy, M. Xu, J. Raiman, M. Shoeybi, B. Catanzaro, and W. Ping. 2025. *NV-Embed: Improved Techniques for Training LLMs as Generalist Embedding Models*. (2025). arXiv: 2405.17428 (cs.CL).

C.-H. Lee, H. Cheng, and M. Ostendorf. 2024. "OrchestraLLM: Efficient Orchestration of Language Models for Dialogue State Tracking." In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by K. Duh, H. Gomez, and S. Bethard. Association for Computational Linguistics, Mexico City, Mexico, 1434–1445. doi:10.18653/v1/2024.naacl-long.79.

S. Lee, D. B. Lee, D. Wagner, M. Kang, H. Seong, T. Bocklet, J. Lee, and S. J. Hwang. 2025. *SafeRoute: Adaptive Model Selection for Efficient and Accurate Safety Guardrails in Large Language Models*. (2025). arXiv: 2502.12464 (cs.CL).

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2020. "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*. Ed. by D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault. Online, 7871–7880. doi:10.18653/v1/2020.acl-main.703.

P. Lewis et al.. 2020. "Retrieval-augmented generation for knowledge-intensive NLP tasks." In: *Proceedings of the 34th International Conference on Neural Information Processing Systems* Article 793. Curran Associates Inc., Red Hook, NY, USA, 9459–9474. https://dl.acm.org/doi/abs/10.5555/3495724.3496517.

C. Li, M. Qin, S. Xiao, J. Chen, K. Luo, D. Lian, Y. Shao, and Z. Liu. 2025. "Making Text Embedders Few-Shot Learners." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=wfLuiDjQ0u.

Y. Li. 2025. *LLM Bandit: Cost-Efficient LLM Generation via Preference-Conditioned Dynamic Routing*. (2025). arXiv: 2502.02743 (cs.LG).

Z. Li, C. Li, M. Zhang, Q. Mei, and M. Bendersky. Nov. 2024. "Retrieval Augmented Generation or Long-Context LLMs? A Comprehensive Study and Hybrid Approach." In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Ed. by F. Dernoncourt, D. Preoţiuc-Pietro, and A. Shimorina. Association for Computational Linguistics, Miami, Florida, US, (Nov. 2024), 881–893. doi:10.18653/v1/2024.emnlp-industry.66.

C.-C. Lin, A. Y. Q. Huang, and S. J. H. Yang. 2023. "A Review of AI-Driven Conversational Chatbots Implementation Methodologies and Challenges (1999–2022)." *Sustainability*, 15, 5, 1–13. doi:10.3390/su15054012.

C.-Y. Lin. 2004. "ROUGE: A Package for Automatic Evaluation of Summaries." In: *Text Summarization Branches Out*. Association for Computational Linguistics, Barcelona, Spain, 74–81. https://aclanthology.org/W04-1013/.

L. Lin et al.. June 2025. "Against The Achilles' Heel: A Survey on Red Teaming for Generative Models." *J. Artif. Int. Res.*, 82, (June 2025), 89 pages. doi:10.1613/jair.1.17654.

Q. Lin, X. Ji, S. Zhai, Q. Shen, Z. Zhang, Y. Fang, and Y. Gao. 2025. *Life-Cycle Routing Vulnerabilities of LLM Router*. (2025). arXiv: 2503.08704 (cs.CR).

J. Liu, R. Gong, M. Zhang, Y. He, J. Cai, and B. Zhuang. 2024. *ME-Switch: A Memory-Efficient Expert Switching Framework for Large Language Models*. (2024). arXiv: 2406.09041 (cs.CL).

S. Liu et al.. 2025. "Rethinking machine unlearning for large language models." *Nature Machine Intelligence*, 7, 181–194. doi:10.1038/s42256-025-00985-0.

Y. Liu, H. Zhang, Y. Miao, V.-H. Le, and Z. Li. 2024. *OptLLM: Optimal Assignment of Queries to Large Language Models*. (2024). arXiv: 2405.15130 (cs.SE).

Z. Liu, C. Li, S. Xiao, Y. Shao, and D. Lian. 2024. "Llama2Vec: Unsupervised Adaptation of Large Language Models for Dense Retrieval." In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by L.-W. Ku, A. Martins, and V. Srikumar. Association for Computational Linguistics, Bangkok, Thailand, 3490–3500. doi:10.18653/v1/2024.acl-long.191.

K. Lu, H. Yuan, R. Lin, J. Lin, Z. Yuan, C. Zhou, and J. Zhou. 2024. "Routing to the Expert: Efficient Reward-guided Ensemble of Large Language Models." In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by K. Duh, H. Gomez, and S. Bethard. Association for Computational Linguistics, Mexico City, Mexico, 1964–1974. doi:10.18653/v1/2024.naacl-long.109.

S. Luccioni, B. Trevelin, and M. Mitchell. 2024. *The Environmental Impacts of AI – Policy Primer*. (2024). doi:10.57967/hf/3004.

X. Ma, L. Wang, N. Yang, F. Wei, and J. Lin. 2024. "Fine-Tuning LLaMA for Multi-Stage Text Retrieval." In: *SIGIR '24: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2421–2425. ISBN: 9798400704314. doi:10.1145/3626772.3657951.

M. Malekpour, N. Shaheen, F. Khomh, and A. Mhedhbi. 2024. "Towards Optimizing SQL Generation via LLM Routing." In: *NeurIPS 2024 Third Table Representation Learning Workshop*. https://openreview.net/forum?id=VYvYR7U7s3.

D. M. Manias, A. Chouman, and A. Shami. 2024. *Semantic Routing for Enhanced Performance of LLM-Assisted Intent-Based 5G Core Network Management and Orchestration*. (2024). arXiv: 2404.15869 (cs.NI).

A. Mohammadshahi, A. Shaikh, and M. Yazdani. 2024. *Routoo: Learning to Route to Large Language Models Effectively*. (2024). arXiv: 2401.13979 (cs.CL).

N. Muennighoff, H. SU, L. Wang, N. Yang, F. Wei, T. Yu, A. Singh, and D. Kiela. 2025. "Generative Representational Instruction Tuning." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=BC4lIvfSzv.

K. S. Narendra and M. A. L. Thathachar. 1974. "Learning Automata - A Survey." *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-4, 4, 323–334. doi:10.1109/TSMC.1974.5408453.

Q. H. Nguyen, D. C. Hoang, J. Decugis, S. Manchanda, N. V. Chawla, and K. D. Doan. 2024. *MetaLLM: A High-performant and Cost-efficient Dynamic Framework for Wrapping LLMs*. (2024). arXiv: 2407.10834 (cs.LG).

X. Ning, Z. Lin, Z. Zhou, Z. Wang, H. Yang, and Y. Wang. 2024. "Skeleton-of-Thought: Prompting LLMs for Efficient Parallel Generation." In: *The 12th International Conference on Learning Representations*. https://openreview.net/forum?id=mqVgBbNCm9.

I. Ong, A. Almahairi, V. Wu, W.-L. Chiang, T. Wu, J. E. Gonzalez, M. W. Kadous, and I. Stoica. 2025. "RouteLLM: Learning to Route LLMs from Preference Data." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=8sSqNntaMr.

S. G. Patil, T. Zhang, X. Wang, and J. E. Gonzalez. 2024. "Gorilla: Large Language Model Connected with Massive APIs." In: *The 38th Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=tBRNC6YemY.

J. Pichlmeier, P. Ross, and A. Luckow. 2024. *Performance Characterization of Expert Router for Scalable LLM Inference*. (2024). arXiv: 2404.15153 (cs.CL).

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al.. 2019. *Language models are unsupervised multitask learners*. (2019). https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. "Exploring the limits of transfer learning with a unified text-to-text transformer." *Journal of Machine Learning Research*, 21, 1, 1–67. http://jmlr.org/papers/v21/20-074.html.

G. Ramírez, A. Birch, and I. Titov. 2024. "Optimising Calls to Large Language Models with Uncertainty-Based Two-Tier Selection." In: *First Conference on Language Modeling*. https://openreview.net/forum?id=T9cOYH0wGF.

M. Sakota, M. Peyrard, and R. West. 2024. "Fly-Swat or Cannon? Cost-Effective Language Model Choice via Meta-Modeling." In: *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 606–615. doi:10.1145/3616855.3635825.

V. Sanh, L. Debut, J. Chaumond, and T. Wolf. 2020. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter." *CoRR*, cs.CL/1910.01108v4. https://arxiv.org/abs/1910.01108.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. 2017. *Proximal Policy Optimization Algorithms*. (2017). arXiv: 1707.06347 (cs.LG).

A. Shafran, R. Schuster, T. Ristenpart, and V. Shmatikov. 2025. "Rerouting LLM Routers." In: *Second Conference on Language Modeling*. https://openreview.net/forum?id=U6C7odo5SX.

Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang. 2023. "HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face." In: *37th Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=yHdTscY6Ci.

T. Shnitzer, A. Ou, M. Silva, K. Soule, Y. Sun, J. Solomon, N. Thompson, and M. Yurochkin. 2024. *Large Language Model Routing with Benchmark Datasets*. (2024). arXiv: 2309.15789 (cs.CL).

C. Si, W. Shi, C. Zhao, L. Zettlemoyer, and J. Boyd-Graber. 2023. "Getting MoRE out of Mixture of Language Model Reasoning Experts." In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by H. Bouamor, J. Pino, and K. Bali. Association for Computational Linguistics, Singapore, 8234–8249. doi:10.18653/v1/2023.findings-emnlp.552.

D. Sikeridis, D. Ramdass, and P. Pareek. 2024. *PickLLM: Context-Aware RL-Assisted Large Language Model Routing*. (2024). arXiv: 2412.12170 (cs.LG).

T. Simonds, K. Kurniawan, and J. H. Lau. 2024. "MoDEM: Mixture of Domain Expert Models." In: *Proceedings of the 22nd Annual Workshop of the Australasian Language Technology Association*. Ed. by T. Baldwin, S. J. Rodríguez Méndez, and N. Kuo, 75–88. https://aclanthology.org/2024.alta-1.6/.

S. Somerstep, F. M. Polo, A. F. M. de Oliveira, P. Mangal, M. Silva, O. Bhardwaj, M. Yurochkin, and S. Maity. 2025. *CARROT: A Cost Aware Rate Optimal Router*. (2025). arXiv: 2502.03261 (stat.ML).

K. A. Srivatsa, K. Maurya, and E. Kochmar. 2024. "Harnessing the Power of Multiple Minds: Lessons Learned from LLM Routing." In: *Proceedings of the Fifth Workshop on Insights from Negative Results in NLP*. Ed. by S. Tafreshi, A. Akula, J. Sedoc, A. Drozd, A. Rogers, and A. Rumshisky. Association for Computational Linguistics, Mexico City, Mexico, 124–134. doi:10.18653/v1/2024.insights-1.15.

D. Stripelis et al.. 2024. "TensorOpera Router: A Multi-Model Router for Efficient LLM Inference." In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Ed. by F. Dernoncourt, D. Preoțiuc-Pietro, and A. Shimorina. Association for Computational Linguistics, Miami, Florida, US, 452–462. doi:10.18653/v1/2024.emnlp-industry.34.

R. S. Sutton and A. G. Barto. 2014. *Temporal-Difference Learning*. The MIT Press. Chap. 6, 143–166. https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf.

H. Touvron et al.. 2023. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. (2023). arXiv: 2307.09288 (cs.CL).

I. Turc, M. Chang, K. Lee, and K. Toutanova. 2019. *Well-Read Students Learn Better: The Impact of Student Initialization on Knowledge Distillation*. (2019). arXiv: 1908.08962 (cs.CL).

C. Wang, B. Zhang, D. Sui, Z. Tu, X. Liu, and J. Kang. 2024. *A survey on effective invocation methods of massive LLM services*. (2024). arXiv: 2402.03408 (cs.SE).

H. Wang, F. M. Polo, Y. Sun, S. Kundu, E. Xing, and M. Yurochkin. 2024. "Fusing Models with Complementary Expertise." In: *The 12th International Conference on Learning Representations*. https://openreview.net/forum?id=PhMrGCMIRL.

L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei. Aug. 2024. "Improving Text Embeddings with Large Language Models." In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by L.-W. Ku, A. Martins, and V. Srikumar. Association for Computational Linguistics, Bangkok, Thailand, (Aug. 2024), 11897–11916. doi:10.18653/v1/2024.acl-long.642.

Q. Wang, J. P. Zhou, Z. Zhou, S. Shin, B. Han, and K. Q. Weinberger. 2025. "Rethinking LLM Unlearning Objectives: A Gradient Perspective and Go Beyond." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=huo8MqVH6t.

X. Wang, Y. Liu, W. Cheng, X. Zhao, Z. Chen, W. Yu, Y. Fu, and H. Chen. 2025. *MixLLM: Dynamic Routing in Mixed Large Language Models*. (2025). arXiv: 2502.18482 (cs.CL).

Y. Wang, X. Zhang, J. Zhao, S. Wen, P. Feng, S. Liao, L. Huang, and W. Wu. 2024. *Bench-CoE: a Framework for Collaboration of Experts from Benchmark*. (2024). arXiv: 2412.04167 (cs.AI).

H. Wei, S. He, T. Xia, A. Wong, J. Lin, and M. Han. 2024. *Systematic Evaluation of LLM-as-a-Judge in LLM Alignment Tasks: Explainable Metrics and Diverse Prompt Templates*. (2024). arXiv: 2408.13006 (cs.CL).

J. Wei et al.. 2022. "Emergent Abilities of Large Language Models." *Transactions on Machine Learning Research*. https://openreview.net/forum?id=yzkSU5zdwD.

C. Xiao et al.. 2025. "Densing law of LLMs." *Nature Machine Intelligence*, 7, 1823–1833. doi:10.1038/s42256-025-01137-0.

M. Xiong, Z. Hu, X. Lu, Y. LI, J. Fu, J. He, and B. Hooi. 2024. "Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs." In: *The 12th International Conference on Learning Representations*. https://openreview.net/forum?id=gjeQKFxFpZ.

H. Xu, T. Zhu, L. Zhang, W. Zhou, and P. S. Yu. Aug. 2023. "Machine Unlearning: A Survey." *ACM Comput. Surv.*, 56, 1, Article 9, (Aug. 2023), 36 pages. doi:10.1145/3603620.

A. Yang et al.. 2024. *Qwen2 Technical Report*. (2024). arXiv: 2407.10671 (cs.CL).

S. Ye, D. Kim, S. Kim, H. Hwang, S. Kim, Y. Jo, J. Thorne, J. Kim, and M. Seo. 2024. "FLASK: Fine-grained Language Model Evaluation based on Alignment Skill Sets." In: *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=CYmF38ysDa.

J. Yuan et al.. 2025. "Who Routes the Router: Rethinking the Evaluation of LLM Routing Systems." In: *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*. https://openreview.net/forum?id=EEPostHMtF.

W. Yuan, G. Neubig, and P. Liu. 2021. "BARTScore: Evaluating Generated Text as Text Generation." In: *Advances in Neural Information Processing Systems*. https://openreview.net/forum?id=5Ya8PbvpZ9.

M. Yue, J. Zhao, M. Zhang, L. Du, and Z. Yao. 2024. "Large Language Model Cascades with Mixture of Thought Representations for Cost-Efficient Reasoning." In: *The 12th International Conference on Learning Representations*. https://openreview.net/forum?id=6okaSfANzh.

J. Zhang, R. Krishna, A. H. Awadallah, and C. Wang. 2023. *EcoAssistant: Using LLM Assistant More Affordably and Accurately*. (2023). arXiv: 2310.03046 (cs.SE).

L. Zhang, K. Jijo, S. Setty, E. Chung, F. Javid, N. Vidra, and T. Clifford. 2024. *Enhancing Large Language Model Performance To Answer Questions and Extract Information More Accurately*. (2024). arXiv: 2402.01722 (cs.CL).

T. Zhang, A. Mehradfar, D. Dimitriadis, and S. Avestimehr. 2025. *Leveraging Uncertainty Estimation for Efficient LLM Routing*. (2025). arXiv: 2502.11021 (cs.NI).

Y. Zhang. 2022. *An Introduction to Matrix factorization and Factorization Machines in Recommendation System, and Beyond.* (2022). arXiv: 2203.11026 (cs.IR).
P. Zhao, W. Zhu, P. Jiao, D. Gao, and O. Wu. 2025. *Data Poisoning in Deep Learning: A Survey.* (2025). arXiv: 2503.22759 (cs.CR).
Z. Zhao, S. Jin, and Z. M. Mao. 2024. *Eagle: Efficient Training-Free Router for Multi-LLM Inference.* (2024). arXiv: 2409.15518 (cs.LG).
Z. Zhao, L. Gan, G. Wang, W. Zhou, H. Yang, K. Kuang, and F. Wu. 2024. "LoraRetriever: Input-Aware LoRA Retrieval and Composition for Mixed Tasks in the Wild." In: *ACL (Findings)*, 4447–4462. https://doi.org/10.18653/v1/2024.findings-acl.263.
S. Zhou, C. Liu, D. Ye, T. Zhu, W. Zhou, and P. S. Yu. Dec. 2022. "Adversarial Attacks and Defenses in Deep Learning: From a Perspective of Cybersecurity." *ACM Comput. Surv.*, 55, 8, Article 163, (Dec. 2022), 39 pages. doi:10.1145/3547330.
L. Zhuang, L. Wayne, S. Ya, and Z. Jun. 2021. "A Robustly Optimized BERT Pre-training Approach with Post-training." In: *Proceedings of the 20th Chinese National Conference on Computational Linguistics*. Ed. by S. Li, M. Sun, Y. Liu, H. Wu, K. Liu, W. Che, S. He, and G. Rao, 1218–1227. https://aclanthology.org/2021.ccl-1.108/.
R. Zhuang, T. Wu, Z. Wen, A. Li, J. Jiao, and K. Ramchandran. 2025. "EmbedLLM: Learning Compact Representations of Large Language Models." In: *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=Fs9EabmQrJ.

A Descriptive table of the routing strategies

Table 2. A description of the various works included in the survey

Study	Task	Benchmarks	Compared Strategies	Routing Step	Routing Approach
Aggarwal et al. (2024)	Short and Multi-Choice Q/A, Text understanding, Reasoning	QASPER, QUALITY, COQA, MUTUAL, DIPLOMAT	Stand-alone models, FrugalGPT (L. Chen et al. 2023), HybridLLM ((Ding et al. 2024))	Post	Repeated calls
Chai et al. (2024)	Multi-domain Q/A	MMLU Expert	Prompting methods, gold routing, LLM-Blender	Pre	LLM Fine-Tuning
L. Chen et al. (2023)	Short Q/A, Text classification	HEADLINES, OVERRULING, COQA	Stand-alone LLM, top LLM	Post	Answer Confidence Inference
S. Chen et al. (2024)	Multi-domain and Multi-Choice Q/A, Code generation	MMLU, GSM8K, CMMLU, ARC-Challenge, HumanEval, PreAlgebra, MBPP, C-EVAL	Stand-alone models, majority voting, ZOOTER (Lu et al. 2024), multi-class classification and clustering	Pre	Query Similarity
Chuang et al. (2024)	Multi-domain and Multi-Choice Q/A, Code generation	MMLU, OpenbookQA, GSM8K, MedQA	Verbalised confidence, logits-based uncertainty, random routing	Post	LLM Fine-Tuning
Dekoninck et al. (2025) (1)	Multi-domain and Multi-Choice Q/A	ARC-Challenge, MMLU-Pro, MixEval, GSM8k	Linear interpolation, linear optimisation programs (query-based classification, cascading)	Post	Answer Confidence Inference
Dekoninck et al. (2025) (2)	Multi-domain and Multi-Choice Q/A	ARC-Challenge, MMLU-Pro, MixEval, GSM8k	Linear interpolation, linear optimisation programs (query-based classification, cascading)	Pre	Query Complexity Inference
Ding et al. (2024)	Instructions	MixInstruct	Random routing, smallest LLM, largest LLM	Pre	Query Complexity Inference

Study	Task	Benchmarks	Compared Strategies	Routing Step	Routing Approach
Feng et al. (2025)	Multi-domain Q/A, Text understanding, Summarization	Alpaca, GSM8K, SQUAD, MultiNews	FrugalGPT (L. Chen et al. 2023), HybridLLM (Ding et al. 2024), prompt routing, smallest LLM, largest LLM, gold routing, bandit-based model	Pre	Knowledge Graph
Hari and Thomson (2023)	Multi-Domain Text Corpus	Expert corpus (e.g., Pile-CC, Pubmed Central, ArXiv)	Stand-alone models, Gorilla ((Patil et al. 2024))	Pre	Reward-based Inference
Q. J. Hu et al. (2024)	Multi-domain and Multi-Choice Q/A, Instructions, Reasoning	MMLU, MT-Bench, MBPP, HellaSwag, Winogrande, GSM8K, Arc-Challenge	Stand-alone models, K-NN, MLP, linear interpolation, oracle routing	Pre	Query Complexity Inference
Jain et al. (2024)	Multi-domain Q/A	MMLU Pro, GSM8K	Stand-alone models	Pre	Domain Classification
Jang et al. (2023)	Q/A, Reasoning, Text understanding, Text classification, Instructions	RTE, CB, ANLI, COPA, HellaSWAG, Storycloze, WinoGrande, WSC, WiC, Big-Bench, wiki-auto, HGen, COVID-QA, ELI5	Stand-alone models	Pre	Query Similarity
Jeong et al. (2024)	Q/A, Text understanding	Single-step Q/A (e.G., SQuAD-v1.1, TriviaQA), Multi-step Q/A (e.g., MuSiQue, HotpotQA)	Adaptive Retrieval, Self-RAG, gold routing	Pre	Query Complexity Inference
Jitkrittum et al. (2025)	Multi-Domain and Multi-Choice Q/A, Code generation, Reasoning, Instructions	EmbedLLM, RouterBench, Mix-Instruct	Random, KNN	Pre	Clustering Previous Interactions
C.-H. Lee et al. (2024) (1)	DST	MultiWOZ, SGD	Prompt-DST, IC-DST and DS2-T5	Pre	Query Similarity
C.-H. Lee et al. (2024) (2)	DST	MultiWOZ, SGD	Prompt-DST, IC-DST and DS2-T5	Post	Sequence Probability
S. Lee et al. (2025)	Adversarial attack detection	HarmBench, OpenAI Moderation, ToxicChat, WildGuardMix, XSTest	Uncertainty (e.g. entropy)	Pre	Query Complexity Inference
Z. Li et al. (2024)	Q/A, Fact extraction, Text understanding	NarrativeQA, QASPER, MultiFieldQA, HotpotQA,etc.	Naïve RAG, long context	Post	Prompt-based Routing

Study	Task	Benchmarks	Compared Strategies	Routing Step	Routing Approach
J. Liu et al. (2024)	Multi-domain and Multi-choice Q/A, Code generation	MMLU, GSM8K, MATH, HumanEval, MBPP, C-Eval and C-MMLU	Stand-alone models	Pre	LLM Fine-Tuning
Lu et al. (2024)	Q/A, Instructions, Multi-domain Q/A, Code generation	AlpacaEval, FLASK, MT-Bench, MMLU, GSM8K, HumanEval	Stand-alone models, overall best LLM, oracle routing	Pre	Reward-based Inference
Malekpour et al. (2024)	Text-to-SQL	BIRD	Stand-alone models	Pre	Query Similarity, Query complexty inference
Manias et al. (2024)	Intent detection	Custom Dataset	/	Pre	Query Similarity
Mohammadshahi et al. (2024)	Multi-Domain Q/A	MMLU	Stand-alone models	Pre	Decoder-based encoding
Nguyen et al. (2024)	Text classification	IMDB, SST-2	Stand-alone models	Pre	State-based algorithms
Ning et al. (2024)	Q/A	FastChat, LLMZoo	No routing	Pre	Query Complexity Inference
Ning et al. (2024)	Q/A	FastChat, LLMZoo	No routing	Pre	Prompt-based Routing
Ong et al. (2025)	Multi-Domain Q/A, Instructions	MT Bench, MMLU, GSM8K	Random routing	Pre	Query Complexity Inference, Preference Similarity, Recommendation System
Ong et al. (2025)	Multi-Domain Q/A, Instructions	MT Bench, MMLU, GSM8K	Random routing	Pre	LLM Fine-Tuning
Patil et al. (2024)	Code generation	Custom API calling dataset	Stand-alone models	Pre	LLM Fine-Tuning
Pichlmeier et al. (2024)	Feasibility	/	/	Pre	Clustering Previous Interactions
Ramírez et al. (2024)	Text classification, Multi-Domain and Multi-Choice Q/A, Fact Extraction	Wikifact, Openbook, ISEAR, FEVER, bAbI, Natural Questions, SST-2, CR, RT-Polarity	FrugalGPT (L. Chen et al. 2023), HybridLLM (Ding et al. 2024), query-based classification	Post	Token Probability
Sakota et al. (2024)	Text classification, Multi-Domain and Multi-Choice Q/A	MMLU, GSM8K, WikiFact, RAFT, LegalSupport, etc.	Oracle routing, stand-alone models	Pre	Query Complexity Inference
Shen et al. (2023)	Task Planning	Custom Dataset	/	Pre	Prompt-based Routing
Shnitzer et al. (2024)	Multi-Domain Q/A, Instructions, Text classification	HELM, MixInstruct	Stand-alone models, overall best LLM, oracle routing,	Pre	Query Complexity Inference

Study	Task	Benchmarks	Compared Strategies	Routing Step	Routing Approach
Sikeridis et al. (2024)	Q/A	HC3	Random routing, stand-alone models	Pre	Stateless Algorithms
Simonds et al. (2024)	Multi-Domain and Multi-Choice Q/A, Code generation	MMLU, MMLU Pro, GPQA, HumanEval, College Math, MATH, GSM8k, Olympiad Bench	Stand-alone models	Pre	Domain Classification
Somerstep et al. (2025)	Multi-Domain and Multi-Choice Q/A, Code generation, Instructions, Reasoning	IFEval, GSM8K, MMLU, Arc-Challenge, MBPP, MT-Bench, WinoGrande, HellaSWAG, MMLU-Pro, GPQA, MATH, BigBench-Hard, MuSR, OpenHermes-2.5, RAGBench	Random forests, RouteLLM (Ong et al. 2025)	Pre	Query Complexity Inference
Srivatsa et al. (2024)	Multi-domain Q/A	MMLU, GSM8K	Stand-alone models, random routing, oracle routing	Pre	Clustering Previous Interactions, Query Complexity Inference
Stripelis et al. (2024)	Multi-domain Q/A, Code generation	Ai2-ARC, GSM8K, MBPP, PubMedQA	Random routing, stand-alone models, oracle routing	Pre	Query Similarity, Query Complexity Inference
Y. Wang et al. (2024)	Multi-domain and Multi-Choice Q/A, Code generation, Reasoning	MMLU Pro, GSM8K, Winogrande, Big Bench Hard, MMMU, MMStar	Stand-alone models	Pre	Domain Classification, Query Complexity Inference
Yue et al. (2024)	Multi-Domain Q/A, Text classification, Reasoning, Text understanding	GSM8K, ASDIV, TabMWP, Big-Bench Hard, CREPE	Stand-alone models	Pre	Query Complexity Inference
J. Zhang et al. (2023)	Instructions	ToolBench	Stand-alone models	Post	Code Execution
T. Zhang et al. (2025)	Multi-Choice and Multi-Domain Q/A	Natural Questions, TriviaQA, MAWPS, POPQA	TensorOpera (Stripelis et al. 2024), RouteLLM (Ong et al. 2025)	Pre	Recommendation Dystem
Z. Zhao, Jin, et al. (2024)	Multi-Domain and Multi-Choice Q/A, Code generation , Instructions	MMLU, Hellaswag, GSM8K, ARC-Challenge, Winogrande, MBPP, MT-Bench	Linear SVM, KNN and MLP	Pre	Preference Dimilarity

Study	Task	Benchmarks	Compared Strategies	Routing Step	Routing Approach
R. Zhuang et al. (2025)	Multi-Choice and Multi-Domain Q/A, Code generation	MMLU, GSM8K, GPQA, TruthfulQA, ASDIV, LogiQA, MathQA, MedMCQA, PIQA, SocialQA	Best Model	Pre	Recommendation Dystem

Received 09 July 2025; accepted 19 February 2026