ESP IJAST
ESP International Journal of Advancements in Science & Technology
ISSN: 2583-9233/ Volume 4 Issue 4 October-December 2026
Paper Id: IJAST-V4I4P103/Page No: 16-24

Original Article

A Survey of Prompt Engineering Techniques for Large Language Models: A Comparative Summary

Surenda Kumawat[1], Dr. Vishal Shrivastava[2], Dr. Akhil Pandey[3]
[1,2,3]*Computer Science & Engineering, Arya College of Engineering & I.T. Jaipur, India*

Received Date: 15 October 2026 Revised Date: 20 October 2026 Accepted Date: 25 October 2026

Abstract: *Prompt engineering has emerged as the primary interface through which practitioners steer the behaviour of large language models (LLMs) without updating any model weights. Since the introduction of few-shot in-context learning, a wide family of prompting strategies has been proposed, ranging from simple zero-shot instructions to elaborate reasoning frameworks and agentic, tool-using prompts. This paper presents a comparative survey of the major prompt engineering techniques reported in the literature between 2020 and 2024, organising them into four broad families: demonstration-based prompting (zero-shot and few-shot learning), reasoning-structured prompting (chain-of-thought, self-consistency, tree- and graph-of-thoughts), agentic and tool-augmented prompting (ReAct and reflection-based methods), and optimization-based prompting (automatic prompt engineering and soft/prefix prompt tuning). For each family we summarise representative studies, the benchmarks used to evaluate them, and their reported strengths and limitations. We further propose a comparative framework built around six evaluation criteria – accuracy uplift, inference cost, human engineering effort, robustness, interpretability, and applicability under black-box (API-only) access – and use it to place the surveyed techniques on a common footing. Reported benchmark figures collected from the literature (e.g., chain-of-thought with self-consistency raising PaLM-540B accuracy on GSM8K from 56.5% to 74.4%, and Tree-of-Thoughts raising GPT-4's success rate on the Game of 24 from 4% to 74%) illustrate that large accuracy gains are consistently accompanied by proportionally larger inference-time costs. The survey concludes that no single technique dominates across all criteria, and that the choice of prompting strategy should be guided by task difficulty, latency and cost budgets, and the degree of interpretability required in a given deployment.*

Keywords: *Large Language Models, Prompt Engineering, Chain-Of-Thought, In-Context Learning, Tree of Thoughts, React, Self-Consistency, Automatic Prompt Engineering, Comparative Survey.*

I. INTRODUCTION

Large language models (LLMs) such as GPT-3, PaLM, LLaMA and GPT-4 have demonstrated that a single pre-trained model can be adapted to a very large number of downstream tasks purely through natural-language instructions supplied at inference time, a capability commonly referred to as in-context learning. This departs sharply from the earlier pre-train-then-fine-tune paradigm, in which every new task required updating some or all of a model's parameters. Because the input prompt is now the principal lever available to a practitioner, the way a prompt is worded, structured, and sequenced has a large and often non-obvious effect on output quality. The practice of deliberately designing, testing and refining these inputs has come to be known as prompt engineering.

Prompt engineering research has grown extremely quickly since 2020, producing a large and fragmented body of techniques with different goals: some techniques aim purely to demonstrate a task through examples, others aim to elicit multi-step reasoning, others allow the model to interact with external tools, and others try to automate the discovery of a good prompt altogether. Because these techniques are evaluated on different benchmarks, with different base models and different reporting conventions, it is difficult for a newcomer to compare them on a common footing. This motivates the present survey, whose goal is not to introduce a new prompting technique, but to organise the existing literature into a clear taxonomy, summarise the reported evidence for each technique's effectiveness, and build a comparative framework that allows the techniques to be weighed against one another along several practical dimensions rather than accuracy alone.

A. Large Language Models and In-Context Learning

Modern LLMs are transformer-based autoregressive models trained on very large text corpora using a next-token-prediction objective. Once trained at sufficient scale, such models exhibit in-context learning: when a handful of input-output examples are placed in the prompt, the model can infer the underlying task and apply it to a new query without any gradient update. This ability was first demonstrated at scale with GPT-3 and has since been confirmed across many model families. In-context learning is the foundation on which nearly all subsequent prompt engineering techniques are built, because it establishes that the prompt itself, rather than the model's weights, can encode task-specific behaviour.

This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/2.0/)

B. Motivation for a Comparative Summary

A practitioner choosing a prompting strategy for a real application must typically balance several competing concerns: the accuracy gain a technique offers over a naive baseline, the additional inference cost it introduces (extra tokens, repeated sampling, or external tool calls), the amount of manual effort required to design exemplars or scoring functions, how sensitive the technique is to small wording changes, whether it produces an inspectable reasoning trace, and whether it can be used purely through a black-box API or requires access to model gradients. Individual papers typically report only accuracy on their own chosen benchmark, leaving these other dimensions to be inferred indirectly. The remainder of this paper reviews the major technique families reported in the literature (Section 2), proposes a comparative framework and taxonomy for organising them (Section 3), applies that framework to synthesise the reported benchmark evidence (Section 4), and closes with conclusions and directions for future work (Section 5).

C. Scope and Organization of this Survey

This survey deliberately restricts itself to techniques that act primarily through the prompt supplied to an already pre-trained language model, rather than techniques that require retraining a model from scratch or performing full-parameter fine-tuning on a downstream task. Retrieval-augmented generation, instruction fine-tuning, and reinforcement learning from human feedback are related and influential lines of work, but they are treated only where they intersect directly with a prompting technique discussed here (for example, instruction-tuned models are often the base model on which a prompting technique is subsequently evaluated); a full treatment of these adjacent areas is left outside the scope of the present survey. Within this scope, the paper is organised as follows. Section 2 reviews the literature underlying each of the four technique families identified above and tabulates representative studies together with the benchmarks, strengths, and limitations reported by their authors. Section 3 formalises a comparative framework and a set of six evaluation criteria that are used to place the families on a common footing. Section 4 applies this framework to the evidence gathered in Section 2, using figures reconstructed from the original papers to illustrate the accuracy-versus-cost trade-off empirically, and closes with a set of practical recommendations. Section 5 concludes the survey and outlines directions for future comparative work.

II. RELATED WORKS / LITERATURE SURVEY

With the rapid growth of prompting research since 2020, it is useful to group existing techniques into families that share a common underlying mechanism. Table 1 summarises the taxonomy used throughout this survey; each family is then discussed in a dedicated subsection with representative studies.

Table 1: Taxonomy of Prompt Engineering Technique Families Used in This Survey

Category	Representative Technique(s)	Core Idea	Key Reference(s)
Demonstration-based	Zero-shot, Few-shot / in-context learning	Task is shown through instructions or input–output exemplars placed directly in the prompt	Brown et al. (2020)
Reasoning-structured	Chain-of-Thought, Self-Consistency, Tree/Graph of Thoughts	Model is guided to expose or search over intermediate reasoning steps before answering	Wei et al. (2022); Wang et al. (2022); Yao et al. (2023); Besta et al. (2024)
Agentic / tool-augmented	ReAct, Reflexion / self-refine	Reasoning is interleaved with actions that query external tools or environments	Yao et al. (2022)
Optimization-based	Automatic Prompt Engineer (APE), Soft/Prefix prompt tuning	The prompt itself (discrete text or continuous vectors) is searched for or learned automatically	Zhou et al. (2022); Lester et al. (2021); Li & Liang (2021)

A. Zero-Shot and Few-Shot (Demonstration-Based) Prompting

Brown et al. introduced GPT-3 and showed that a sufficiently large autoregressive language model can perform a new task after being shown only a natural-language instruction (zero-shot) or a small number of input–output pairs (few-shot), without any parameter updates. This finding reframed prompting as a first-class mechanism for task adaptation rather than a minor implementation detail, and it underlies almost every technique discussed later in this survey. Few-shot prompting remains sensitive to the choice, number, and even the ordering of exemplars, which motivated much of the later work on more structured and automatically optimised prompts.

B. Chain-of-Thought and Reasoning-Oriented Prompting

Wei et al. proposed chain-of-thought (CoT) prompting, in which a small number of exemplars in the prompt include a written chain of intermediate reasoning steps leading to the final answer. They reported that CoT produces large improvements on arithmetic, commonsense and symbolic reasoning tasks, and that this benefit emerges mainly at large model

scale, being negligible or even harmful for smaller models. Kojima et al. showed that a similar effect can be obtained without any hand-written exemplars at all, simply by appending the phrase “Let's think step by step” to the prompt, a technique referred to as zero-shot CoT. Because CoT is applied purely at inference time, it requires no fine-tuning, but it also offers no guarantee that the generated reasoning chain is faithful to how the model actually produced its answer, and a single greedily-decoded chain can still propagate an early error to the final answer.

C. Self-Consistency and Ensemble Reasoning

Wang et al. addressed the fragility of a single greedily-decoded reasoning chain by proposing self-consistency, which samples several diverse reasoning paths for the same question using CoT prompting and then selects the final answer that occurs most frequently among the sampled paths. Applied to PaLM-540B, self-consistency raised accuracy on GSM8K from 56.5% (CoT with greedy decoding) to 74.4%, an absolute gain of 17.9 percentage points, with comparable gains reported on SVAMP, AQuA, StrategyQA and ARC-Challenge. The technique requires no additional training and can be layered on top of any CoT prompt, but its cost scales linearly with the number of sampled reasoning paths, since each path requires a full additional generation from the model.

D. Tree-of-Thoughts and Graph-of-Thoughts (Search-Based Prompting)

Yao et al. observed that both plain CoT and self-consistency still commit to reasoning in a single left-to-right pass, which limits their ability to explore alternatives, look ahead, or backtrack. They proposed Tree of Thoughts (ToT), which represents intermediate reasoning steps as nodes in a tree, allows the model to generate and self-evaluate multiple candidate continuations at each step, and applies a classical search algorithm such as breadth-first or depth-first search to explore the tree. On the Game of 24 arithmetic puzzle, GPT-4 with plain CoT solved only 4% of instances, while ToT with a search breadth of five achieved a 74% success rate; ToT also outperformed baselines on Creative Writing and Mini Crosswords tasks that require planning. Besta et al. subsequently generalised this idea to Graph of Thoughts (GoT), which allows reasoning steps to be organised in arbitrary graph topologies rather than a strict tree, enabling operations such as merging or refining previously generated thoughts. Both methods substantially increase inference-time computation relative to a single CoT pass, since many candidate thoughts must be generated and evaluated.

E. ReAct and Agentic, Tool-Augmented Prompting

Yao et al. proposed ReAct, which interleaves chain-of-thought-style reasoning traces with discrete actions that let the model query an external tool or environment, such as a Wikipedia search API. On the multi-hop question answering benchmark HotpotQA and the fact-verification benchmark FEVER, the authors reported that ReAct reduces the hallucination and error-propagation problems observed in pure CoT reasoning, because intermediate claims can be checked against retrieved evidence rather than relying solely on the model's parametric knowledge. On two interactive decision-making benchmarks, ALFWorld and WebShop, ReAct outperformed imitation-learning and reinforcement-learning baselines by an absolute success-rate margin of 34% and 10% respectively, while using only one or two in-context examples. Because ReAct issues explicit tool calls, its latency and monetary cost depend on the number and speed of those external calls, in addition to the cost of the language-model generations themselves.

F. Automatic Prompt Engineering and Prompt Optimization

Rather than relying on a human to hand-craft an effective prompt, Zhou et al. proposed Automatic Prompt Engineer (APE), which treats a natural-language instruction as a “program” to be optimised: an LLM proposes a pool of candidate instructions, each candidate is scored on a held-out set of examples, and the highest-scoring instruction is retained. Across 24 instruction-induction tasks, APE-generated instructions outperformed a prior LLM baseline by a large margin and matched or exceeded human-written instructions on 19 of the 24 tasks; the authors further showed that APE-discovered zero-shot CoT prompts (variants of “Let's think step by step”) could improve on the original hand-written phrase. APE removes the need for manual prompt design, but introduces its own overhead in the form of a candidate-generation and scoring loop that must be run before the optimized prompt can be deployed.

G. Soft Prompting and Prefix Tuning (Parameter-Efficient Prompting)

A related but distinct line of work optimises prompts in continuous embedding space rather than in natural language. Lester et al. proposed prompt tuning, in which a small number of trainable “soft prompt” embeddings are prepended to the input while the rest of the language model is kept entirely frozen; as model scale increases, this lightweight tuning approach was shown to close the gap with full model fine-tuning while updating only a tiny fraction of the parameters. Li and Liang proposed the closely related prefix-tuning, which learns a trainable continuous prefix at every transformer layer rather than only at the input embedding layer, and demonstrated strong parameter-efficient performance on generation tasks such as table-to-text generation and summarisation. Unlike the purely textual techniques described above, these soft-prompting methods require gradient access to the underlying model and therefore cannot be used through a black-box, API-only interface.

Table 2: Representative Reasoning-Oriented and Search-Based Prompting Studies

Benchmark(s)	Technique	Category	Strength	Limitation
GSM8K, SVAMP, AQuA (arithmetic/commonsense reasoning)	Chain-of-Thought (Wei et al., 2022)	Reasoning-structured	Large gains on multi-step reasoning without fine-tuning; effect grows with model scale	Reasoning chain not verified; a single wrong early step propagates to the final answer
Diverse zero-shot reasoning tasks	Zero-Shot-CoT (Kojima et al., 2022)	Reasoning-structured	No hand-written exemplars required	Generally less reliable than few-shot CoT on harder problems
GSM8K, SVAMP, AQuA, StrategyQA, ARC-Challenge	Self-Consistency (Wang et al., 2022)	Reasoning-structured	Large absolute accuracy gains (e.g., +17.9% on GSM8K); no extra training	Cost scales linearly with number of sampled reasoning paths
Game of 24, Creative Writing, Mini Crosswords	Tree of Thoughts (Yao et al., 2023)	Reasoning-structured	Very large gains on tasks needing planning/search (74% vs 4% on Game of 24)	Roughly two orders of magnitude more inference compute than a single CoT pass
General multi-step reasoning tasks	Graph of Thoughts (Besta et al., 2024)	Reasoning-structured	Generalises ToT to arbitrary reasoning topologies (merge/refine thoughts)	Added orchestration and implementation complexity

Table 3: Representative Agentic, Tool-Augmented, and Optimization-Based Prompting Studies

Benchmark(s)	Technique	Category	Strength	Limitation
HotpotQA, FEVER	ReAct (Yao et al., 2022)	Agentic / tool-augmented	Reduces hallucination versus pure CoT by grounding claims in retrieved evidence	Latency/cost depend on number of external tool calls
ALFWorld, WebShop	ReAct (Yao et al., 2022)	Agentic / tool-augmented	+34% and +10% absolute success rate over imitation/RL baselines with only 1–2 examples	Requires a well-designed, task-specific action space
24 instruction-induction tasks	APE (Zhou et al., 2022)	Optimization-based	Matches or beats human-written instructions on 19/24 tasks; removes manual prompt design	Needs a candidate-generation and scoring loop before deployment
Broad NLU/NLG tasks (SuperGLUE, etc.)	Prompt Tuning (Lester et al., 2021)	Optimization-based	Closes the gap with full fine-tuning as model scale grows; tiny parameter footprint	Requires gradient access; not usable through a black-box API
Table-to-text generation, summarisation	Prefix-Tuning (Li & Liang, 2021)	Optimization-based	Parameter-efficient; keeps the backbone model entirely frozen	Needs task-specific training data and gradient access

Taken together, these studies show a general trend: techniques that rely purely on how the prompt text is phrased (demonstration-based and reasoning-structured prompting) are the easiest to deploy, since they require nothing beyond API access, but they leave a substantial amount of performance on the table relative to techniques that either search more exhaustively at inference time (ToT/GoT) or optimise the prompt itself ahead of time (APE, soft prompting). Section 3 formalises this trade-off into a comparative framework.

III. PROPOSED COMPARATIVE FRAMEWORK AND METHODOLOGY

Because the surveyed techniques were originally evaluated on different benchmarks with different base models, a direct numerical ranking is not possible from the literature alone. Instead, this section proposes a qualitative-cum-quantitative comparative framework that places every technique family on a common set of criteria, and uses it in Section 4 to synthesise the benchmark evidence collected in Section 2.

A. Survey and Selection Methodology

Candidate techniques were identified by tracing the citation graph outward from four widely-cited anchor papers – the GPT-3 few-shot learning paper, the chain-of-thought paper, the ReAct paper, and the Automatic Prompt Engineer paper – and retaining techniques that (i) operate purely or primarily through the prompt rather than through full model retraining, (ii) report at least one quantitative benchmark result, and (iii) have been referenced by subsequent survey or empirical work in the prompting literature. This produced the four-family taxonomy of Table 1. Soft/prefix prompt tuning was retained as a boundary case: although it requires gradient access, it is included because it is frequently discussed alongside purely textual prompting methods and provides a useful contrast on the applicability criterion defined below.

B. Taxonomy of Prompt Engineering Techniques

Figure 1 depicts the taxonomy of Table 1 as a tree, grouping individual techniques under the four families used throughout this survey. Reasoning-structured prompting is the most populated family, reflecting the intense research focus on improving multi-step reasoning since the publication of the original chain-of-thought paper.

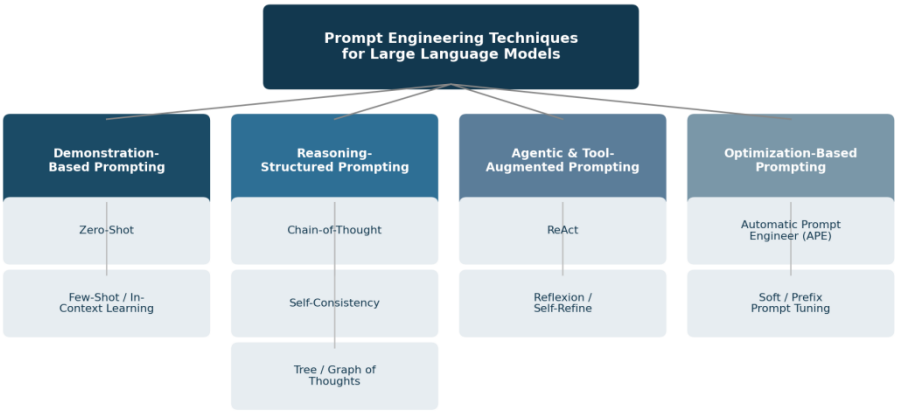


Figure 1: TaxonomyoOf Prompt Engineering Techniques for Large Language Models

C. Comparative Evaluation Criteria

Table 4 defines the six criteria used to compare technique families in Section 4. Accuracy uplift and inference cost are quantified where the underlying papers report the necessary figures; the remaining four criteria are assessed qualitatively based on how each technique is described and used in the literature reviewed in Section 2.

Table 4: Comparative Evaluation Criteria Used in this Survey

Criterion	Description	Illustrative Metric
Accuracy uplift	Improvement over a naive/zero-shot baseline on the same benchmark	Absolute percentage-point accuracy gain
Inference cost	Extra tokens, samples, or tool calls required at inference time relative to a single generation	Number of forward passes / API calls per query
Human engineering effort	Manual effort needed to design exemplars, scoring functions, or action spaces	Qualitative rating: low / medium / high
Robustness	Sensitivity of output quality to prompt wording, ordering, or paraphrase	Qualitative rating based on reported ablations
Interpretability	Whether the technique exposes an inspectable, human-readable reasoning trace	Qualitative rating: yes / partial / no
Applicability	Whether the technique can be used through a black-box API or needs model-internal (gradient) access	Black-box / white-box

D. Comparative Synthesis Procedure

For each technique family, the benchmark figures collected in Section 2 (Tables 2 and 3) were used to populate the accuracy-uplift and inference-cost columns of the synthesis table presented in Section 4 (Table 5); the remaining criteria were assigned based on how the source papers describe the technique's implementation requirements. This procedure does not

claim to produce a single objective ranking – the studies were run on different base models and benchmarks – but it does allow the relative shape of the accuracy/cost trade-off to be compared consistently across families, which is the central contribution of this survey.

IV. COMPARATIVE ANALYSIS AND DISCUSSION

A. Reasoning-Structured Prompting: Accuracy vs. Sampling Cost

Figure 2 plots the GSM8K accuracy of PaLM-540B reported by Wang et al. for chain-of-thought prompting with greedy decoding against the same model with self-consistency decoding layered on top. The 17.9 percentage-point gain is achieved without any additional training, but it is obtained by sampling and marginalising over multiple independent reasoning paths for every query, so the inference cost (and monetary cost, for an API-billed model) scales roughly linearly with the number of samples drawn.

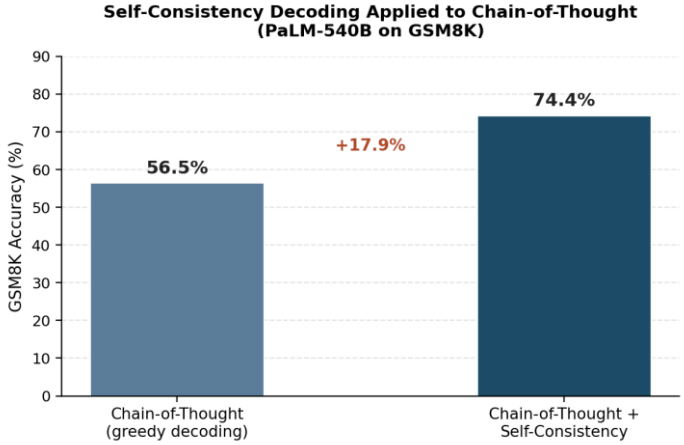


Figure 2: Effect Of Self-Consistency Decoding on Chain-Of-Thought Accuracy (Palm-540B, GSM8K; Data from Wang Et Al., 2022)

B. Search-Based Prompting: Accuracy vs. Search Breadth

Figure 3 compares five prompting strategies evaluated by Yao et al. on the Game of 24 arithmetic puzzle using GPT-4: plain input-output prompting, chain-of-thought, chain-of-thought with self-consistency, and Tree of Thoughts at two search breadths (b = 1 and b = 5). The results show that simply asking the model to reason step by step (CoT) does not help on this particular task – it in fact performs worse than direct input-output prompting – because the task requires exploring and discarding several candidate arithmetic combinations rather than committing to a single line of reasoning. Once the prompting strategy is changed to allow explicit search over multiple candidate thoughts, performance improves sharply, from 9% with self-consistency sampling to 74% with a search breadth of five. This is the clearest illustration in the surveyed literature that reasoning-structured prompting is not a single technique but a spectrum, and that the appropriate point on that spectrum depends heavily on whether the underlying task is fundamentally sequential or fundamentally combinatorial.

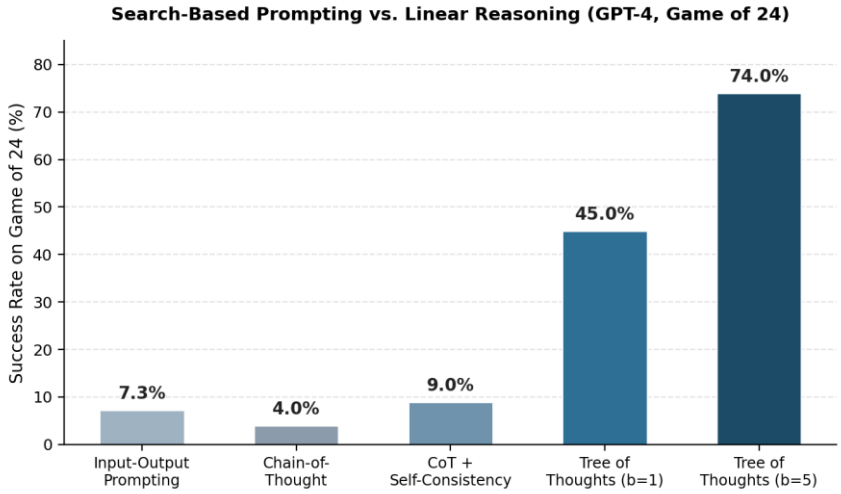


Figure 3: Search-Based Prompting (Tree of Thoughts) Versus Linear Reasoning Strategies (GPT-4, Game Of 24; Data from Yao Et Al., 2023)

C. Agentic and Optimization-Based Prompting

The agentic and optimization-based families reviewed in Section 2 are harder to place on the same numeric axes as Figures 2 and 3 because they were evaluated on different task types (interactive decision-making environments and instruction-induction tasks respectively, rather than single-turn reasoning benchmarks). Nonetheless, the reported figures are informative in relative terms: ReAct's 34-point and 10-point absolute success-rate improvements over imitation/reinforcement-learning baselines on ALFWorld and WebShop show that grounding reasoning in real tool feedback can close gaps that pure text-based reasoning cannot, at the cost of depending on the availability and reliability of the external tools being called. APE's ability to match or beat human-written instructions on 19 of 24 tasks shows that a language model can often replace a human in the prompt-design loop itself, but only after paying the one-time cost of a candidate-generation and scoring search.

D. Synthesis Across All Four Families

Table 5 synthesises the evidence from Sections 2–4 using the criteria defined in Table 4. The general pattern that emerges is a consistent trade-off: moving from demonstration-based prompting toward reasoning-structured, agentic, and finally optimization-based prompting tends to increase achievable accuracy and robustness, but also increases inference cost, human/engineering effort, or both, and progressively reduces the extent to which the technique remains usable through a simple black-box API call.

Table 5: Comparative Synthesis Across the Four Prompt Engineering Technique Families

Family	Typical Accuracy Behaviour	Typical Inference Overhead	Interpretability	Black-Box API Compatible?
Demonstration-based (zero/few-shot)	Baseline; improves with more/better exemplars but plateaus on multi-step tasks	One generation per query (lowest)	No reasoning trace	Yes
Reasoning-structured (CoT/SC/ToT/GoT)	Large gains on reasoning/planning tasks; ToT/GoT gains are largest on search-heavy tasks	Low (CoT) to very high (ToT/GoT, tens of generations)	Partial to yes (explicit chain/tree of thoughts)	Yes
Agentic / tool-augmented (ReAct)	Reduces hallucination; large gains on knowledge- or environment-grounded tasks	Moderate to high (extra tool/API calls per step)	Yes (interleaved reasoning + actions)	Yes, but depends on external tool availability
Optimization-based (APE, soft/prefix tuning)	Matches or exceeds hand-written prompts / full fine-tuning at scale	One-time search or training cost, then low per-query cost	No (soft prompts are not human-readable)	APE: yes; soft/prefix tuning: no (needs gradient access)

Two practical implications follow from this synthesis. First, for latency- or cost-sensitive applications, demonstration-based or single-pass chain-of-thought prompting remains the most economical starting point, and the accuracy gap relative to more expensive techniques is smallest on tasks that are primarily sequential rather than combinatorial. Second, for tasks that plausibly involve search, planning, or verification against external facts – as opposed to a single chain of deduction – the evidence collected in this survey suggests that reasoning-structured search (ToT/GoT) or tool-grounded agentic prompting (ReAct) is likely to be a better investment of inference budget than simply sampling more chain-of-thought paths.

E. Practical Recommendations for Practitioners

Based on the synthesis in Table 5, the following task-oriented recommendations summarise how a practitioner might select an initial prompting strategy before undertaking their own empirical comparison:

- Simple classification, extraction, or single-fact lookup tasks: start with well-chosen few-shot demonstrations; the added complexity of reasoning-structured prompting is rarely justified when no multi-step inference is required.
- Arithmetic, logical, or multi-step word-problem tasks: use chain-of-thought prompting as a baseline, and add self-consistency sampling if the accuracy of a single reasoning chain is insufficient and the added per-query cost is acceptable.
- Planning, combinatorial search, or puzzle-like tasks where a single reasoning chain is prone to dead ends: prefer Tree-of-Thoughts or Graph-of-Thoughts, accepting the substantially higher inference cost in exchange for a much higher success rate.
- Tasks that depend on up-to-date or verifiable external information: prefer ReAct-style, tool-augmented prompting so that intermediate claims can be grounded in retrieved evidence rather than the model's parametric memory alone.

- High-volume, latency-sensitive production deployments where the same task is repeated at scale: consider investing in an offline optimization step (APE or soft/prefix prompt tuning) so that the per-query cost at serving time is minimised, provided gradient access and engineering resources are available for the optimization step.

These recommendations are necessarily heuristic, since – as noted throughout this survey – the underlying studies were conducted on different base models and benchmarks. They are intended as a starting point for technique selection, to be validated empirically on the practitioner's own task and model before being adopted in production.

V. CONCLUSION AND FUTURE WORK

This paper surveyed the major prompt engineering techniques proposed for large language models between 2020 and 2024 and organised them into four families: demonstration-based, reasoning-structured, agentic/tool-augmented, and optimization-based prompting. Reviewing the benchmark evidence reported by the original studies shows that large accuracy gains are consistently available, but that they are not free: self-consistency's 17.9-point gain on GSM8K required sampling multiple reasoning paths per query, and Tree of Thoughts' improvement from 4% to 74% on the Game of 24 required roughly two orders of magnitude more inference-time computation than a single chain-of-thought pass. Agentic, tool-augmented prompting such as ReAct offers a different kind of benefit – grounding reasoning in verifiable external information – at the cost of depending on external tool reliability, while optimization-based techniques such as Automatic Prompt Engineer and soft/prefix prompt tuning shift effort from the deployment phase to a one-time design or training phase, at the cost of reduced human interpretability or the loss of black-box API compatibility.

The comparative framework proposed in Section 3 is necessarily coarse, since it synthesises results collected from heterogeneous studies rather than a single controlled experiment run across all techniques on identical benchmarks and base models. Future work should therefore prioritise standardised, head-to-head benchmarking of representative techniques from each family on a common suite of tasks and a common base model, so that the accuracy-vs-cost trade-offs summarised qualitatively in Table 5 of this survey can be replaced with directly comparable, controlled measurements. Other promising directions include hybrid techniques that combine search-based reasoning with tool grounding, automatic methods for selecting the most cost-appropriate prompting strategy for a given query at run time, and further study of the faithfulness of generated reasoning traces, since several of the studies reviewed here note that an inspectable chain of thought is not guaranteed to reflect the model's true internal computation.

VI. REFERENCES

[1] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., ... Amodei, D. “Language Models are Few-Shot Learners.” Advances in Neural Information Processing Systems (NeurIPS), 2020. arXiv:2005.14165.

[2] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.” Advances in Neural Information Processing Systems (NeurIPS), 2022. arXiv:2201.11903.

[3] Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., & Iwasawa, Y. “Large Language Models are Zero-Shot Reasoners.” Advances in Neural Information Processing Systems (NeurIPS), 2022. arXiv:2205.11916.

[4] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., & Zhou, D. “Self-Consistency Improves Chain of Thought Reasoning in Language Models.” International Conference on Learning Representations (ICLR), 2023. arXiv:2203.11171.

[5] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. “Tree of Thoughts: Deliberate Problem Solving with Large Language Models.” Advances in Neural Information Processing Systems (NeurIPS), 2023. arXiv:2305.10601.

[6] Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., & Hoefler, T. “Graph of Thoughts: Solving Elaborate Problems with Large Language Models.” Proceedings of the AAAI Conference on Artificial Intelligence, 2024. arXiv:2308.09687.

[7] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. “ReAct: Synergizing Reasoning and Acting in Language Models.” International Conference on Learning Representations (ICLR), 2023. arXiv:2210.03629.

[8] Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., & Ba, J. “Large Language Models Are Human-Level Prompt Engineers.” International Conference on Learning Representations (ICLR), 2023. arXiv:2211.01910.

[9] Lester, B., Al-Rfou, R., & Constant, N. “The Power of Scale for Parameter-Efficient Prompt Tuning.” Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2021. arXiv:2104.08691.

[10] Li, X. L., & Liang, P. “Prefix-Tuning: Optimizing Continuous Prompts for Generation.” Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP), 2021. arXiv:2101.00190.

[11] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. “Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing.” ACM Computing Surveys, 55(9), 1–35, 2023. arXiv:2107.13586.

[12] Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q., & Chi, E. “Least-to-Most Prompting Enables Complex Reasoning in Large Language Models.” International Conference on Learning Representations (ICLR), 2023. arXiv:2205.10625.

[13] Zhang, Z., Zhang, A., Li, M., & Smola, A. “Automatic Chain of Thought Prompting in Large Language Models.” International Conference on Learning Representations (ICLR), 2023. arXiv:2210.03493.

[14] White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., & Schmidt, D. C. “A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT.” Proceedings of the 30th Conference on Pattern Languages of Programs (PLoP), 2023.

[15] Yao, S., Chen, H., Yang, J., & Narasimhan, K. “WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents.” Advances in Neural Information Processing Systems (NeurIPS), 2022. arXiv:2207.01206.

[16] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., & Manning, C. D. “HotpotQA: A Dataset for Diverse, Explainable Multi-Hop Question Answering.” Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018.

[17] Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., & Schulman, J. “Training Verifiers to Solve Math Word Problems.” arXiv:2110.14168, 2021.

[18] Liu, X., Zheng, Y., Du, Z., Ding, M., Qian, Y., Yang, Z., & Tang, J. “GPT Understands, Too.” AI Open, 2023 (originally arXiv:2103.10385, 2021).

[19] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. “Training Language Models to Follow Instructions with Human Feedback.” Advances in Neural Information Processing Systems (NeurIPS), 2022. arXiv:2203.02155.