Do LLM Clustering Methods Outperform Classical Baselines? A Controlled, Cost-Aware Benchmark

Adam Zlámal[1]

[1]University of Ostrava, 30. dubna 22, 701 03 Ostrava, Czech Republic

Abstract

Large language models (LLMs) power a growing family of text clustering methods, in which the LLM generates grouping constraints, refines the numerical text representations (embeddings), or summarizes clusters. Published evaluations are hard to compare because each paper uses different datasets, embedding models, and LLMs, and usually reports a single weak baseline such as k-means. We benchmark five LLM-guided clustering methods (Few-Shot keyphrase clustering, k-LLMmeans, SPILL, Cequel, and LSCK-HC) under one controlled protocol: three datasets of short user requests grouped by intent (Banking77, CLINC150, MASSIVE), three embedding models, and one shared LLM. Each method is compared against three classical clustering algorithms run on the same embeddings, for a total of 360 runs: 8 methods (5 LLM-guided and 3 classical) $\times$ 3 datasets $\times$ 3 embedding models $\times$ 5 random seeds. We track clustering accuracy and cost (tokens, US dollars, and running time). Against k-means, the comparison the original papers usually make, the LLM methods often win. Against the strongest classical method in each setting, the gains almost vanish: of the 45 evaluated combinations of method, embedding model, and dataset, only one (SPILL on Banking77 with the instructor-large embedding model, $+6.2 \pm 3.8$ accuracy points) outperforms the best classical algorithm by more than one standard deviation, while requiring one hundred to one thousand times more computation time and money. On strong embedding models, classical algorithms match or outperform every LLM-guided method while requiring no LLM queries at all. The value of LLM-guided clustering declines as embedding quality increases.

Keywords

text clustering, large language models, intent discovery, benchmark, cost-aware evaluation

1. Introduction

Clustering text by meaning underpins intent discovery (grouping user requests by what the user asks for), topic modeling, and document organization. Classical methods such as k-means [1], agglomerative clustering [2], or spectral clustering [3] group texts whose numerical vector representations (embeddings) lie close together. Large language models (LLMs) opened new possibilities for clustering: instead of clustering the embeddings directly, the LLM generates auxiliary signals (constraints, summaries, or neighbor relations) that refine the clustering process [4]. This embedding-refinement approach has produced a proliferation of new methods: Few-Shot Clustering [5] enriches each embedding with keyphrases written by the LLM; k-LLMmeans [6] replaces cluster centers (centroids) with LLM-written cluster summaries; SPILL [7] averages each text's embedding with the embeddings of neighbors the LLM judges to have the same intent; Cequel [8] turns LLM judgments on selected triplets of texts into pairs that must or must not share a cluster; LSCK-HC [9] builds such must-link and cannot-link pairs via LLM classification.

Each paper reports gains over its own baselines, but two questions remain open: *do LLM-based clustering methods outperform classical clustering baselines under a controlled setup? If so, at what cost?* These questions are difficult to answer for four reasons:

- **Different datasets and task variants.** Even when two papers use the same benchmark, they do not necessarily solve the same task: some evaluate on simplified variants that group the texts into a few broad domains instead of the full set of fine-grained intent classes, which makes the task substantially easier; others report accuracy only in figures, so exact numbers cannot be extracted.

ITAT'26: Information Technologies – Applications and Theory, September 25–29, 2026, Biele Karpaty, Slovakia
zlamad07@osu.cz (A. Zlámal)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-09

CEUR-WS.org/Vol-4323/paper22.pdf

- **Different embedding models and LLMs.** Papers pair diverse embedding models with diverse LLMs and prompts, so the effect of the method itself cannot be separated from the strength of the embedding model or the quality of the LLM.
- **Weak baselines.** Papers compare mostly against k-means; agglomerative and spectral clustering (which often outperform k-means on the same embeddings) are rarely reported.
- **Sparse cost reporting.** Token counts, dollar costs, and running times are often omitted, so efficiency tradeoffs cannot be assessed.

We answer these questions with a controlled benchmark: five LLM-guided methods and three classical methods (k-means, Ward agglomerative, and spectral clustering) run on the same test splits of three intent-discovery datasets, the same precomputed embeddings from three embedding models of varying strength, the same LLM, and five random seeds. We report results from 360 runs with full accuracy and cost accounting (Section 3). Our contributions are as follows:

- **A controlled comparison** of five published LLM-guided clustering methods against classical clustering methods under identical data, embedding models, LLM, and random seeds. To our knowledge, this is the first systematic benchmark of this kind.
- **A negative result for LLM-based methods:** we evaluate each method on every combination of embedding model and dataset, 45 comparisons in total. In exactly one of them (SPILL with instructor-large on Banking77, $+6.2 \pm 3.8$ accuracy points) does an LLM-guided method outperform the best classical algorithm by more than one standard deviation; the gains reported in the original papers largely reflect the weakness of the k-means baseline.
- **A cost-aware analysis:** LLM-guided methods require one hundred to one thousand times more time and money than the classical baselines, which run in seconds at negligible cost; on strong embedding models this additional expense does not improve accuracy but reduces it. The benefit of LLM refinement declines as embedding quality increases.
- **A reproducibility study and released repository:** we quantify the gap to published numbers, trace it mainly to the choice of embedding model, and release the code and precomputed embeddings.[1]

2. Methods

2.1. LLM-Guided Clustering via Embedding Refinement

LLM-based clustering spans multiple pipeline architectures, from constraint generation [4] to label generation and taxonomy construction. We reproduce five methods that share a common pattern (Figure 1): the LLM provides a refinement signal fed back to the embedding space, while classical clustering remains the core algorithm. All five share three characteristics: the embedding model is used as-is and never retrained (a *frozen* model); the LLM only provides guidance (keyphrases, summaries, neighbor selections, or pairwise judgments); and classical clustering (k-means or a variant that respects constraints) computes the final clusters.



Figure 1: The shared pipeline: a frozen embedding model turns each text into a vector, the LLM generates a refinement signal from sampled texts, and the signal reshapes the embedding space before classical clustering runs again. The LLM only ever reshapes the embedding space: even where it writes cluster-describing text (k-LLMmeans's summaries), that text is used solely to relocate a centroid, and every method's final output remains an unlabeled cluster assignment: no method attaches a persistent label to a cluster in its result.

[1] https://github.com/jahody/llm-clustering-benchmark

2.2. The Five Tested Methods

Table 1 summarizes the five methods.

Table 1
The five LLM-guided clustering methods compared.

Method	LLM signal	Refinement target	Reference
Few-Shot	intent keyphrases per text	concatenated embeddings	[5]
k-LLMmeans	per-cluster summary sentences	centroid vectors	[6]
SPILL	same-intent neighbor subset	averaged embeddings	[7]
Cequel	triplet judgments	must-link/cannot-link constraints	[8]
LSCK-HC	must-link/cannot-link classification	pairwise constraints	[9]

Few-Shot Clustering. Viswanathan et al. [5] generate intent keyphrases for each text with the LLM; the keyphrase embeddings are concatenated with the text embeddings, and k-means clusters the enriched representation.

k-LLMmeans. Diaz-Rodriguez [6] replace the k-means cluster centers with embeddings of LLM-written cluster summaries; texts are re-assigned each round until the centers stop moving.

SPILL. Lin et al. [7] show each text's nearest neighbors to the LLM, which selects those with the same intent; the selected neighbors' embeddings are averaged with the text's own embedding before clustering.

Cequel. Wang et al. [8] show the LLM three texts at a time (a reference text, one of its close neighbors, and a randomly chosen text) and the LLM decides which of them, if any, express the same intent. The answer labels every pair in the triplet: pairs that share an intent become must-link constraints, the remaining pairs cannot-link constraints, and a variant of k-means that respects these constraints produces the clustering.

LSCK-HC. Jia et al. [9] generate constraints in two stages: small groups of nearby texts are partitioned into must-link sets, and texts are compared against diverse representatives to produce cannot-link pairs; the constraints then drive the constraint-respecting k-means.

3. Experimental Setup

We evaluate 5 methods $\times$ 3 datasets $\times$ 3 embedding models $\times$ 5 random seeds (225 method runs), plus three classical baselines run on the same embeddings and seeds (135 baseline runs), for 360 runs in total. All methods share one LLM, `deepseek-v4-flash` [10]. Throughout, a *setting* denotes one combination of embedding model and dataset (nine in total); each method is evaluated in every setting, with five seeds per setting.

Datasets and embeddings. Table 2 lists the three intent datasets and three embedding models. Each dataset consists of short user texts, each labeled with the intent it expresses. The clustering methods never see these true labels; we use them only to set the number of clusters K and to evaluate the resulting clusters. Following the original papers [7, 8], we cluster the held-out test split of each dataset, with K equal to the number of distinct true labels present in it.

Classical baselines. We compare against three classical algorithms run on the same precomputed embeddings: (1) k-means (Lloyd's algorithm [1], `n_init=10`, the literature standard), (2) agglomerative clustering with Ward linkage [2] (deterministic; it involves no randomness), and (3) spectral clustering (a nearest-neighbor similarity graph, final assignment via k-means) [3], all via scikit-learn [11]. We report two accuracy differences for every LLM-guided method: the difference to k-means, which mirrors the comparisons made in the original papers, and the difference to the strongest of the three classical algorithms in the same setting. The latter is the more demanding test, because a practitioner would simply choose whichever classical algorithm performs best (Section 4.1).

Metrics and cost. We compute clustering accuracy (ACC): the fraction of texts placed in the correct cluster, after the predicted clusters are optimally matched to the true labels with the Hungarian algorithm [12]. We also report normalized mutual information (NMI) and adjusted Rand index (ARI), two standard measures of agreement between a predicted and a true grouping. Each metric is reported as mean $\pm$ standard deviation over the 5 seeds; the paper reports ACC and NMI, with full ARI results in the released repository. We further track input and output tokens, cost in US dollars (USD) at OpenRouter prices (`deepseek-v4-flash`: \$0.09 per million input tokens, \$0.18 per million output tokens), and running time in two variants: *cold*, a first run in which every LLM query is sent to the API and the time spent waiting for responses over the network is included, and *warm*, a repeated run in which all LLM responses are served from a local cache. We report cold-run costs; warm reruns are substantially cheaper because the stored LLM responses are reused.

Hardware. LLM calls and the `qwen3-embedding-8b`/`gemini-embedding-001` embeddings are served over the OpenRouter API, so their wall-clock cost reflects network/API latency, not local hardware. The `instructor-large` embeddings and all classical-baseline clustering (k-means, Ward, spectral) run locally, on a single Windows machine (AMD Ryzen 7 5700G, 16 GB RAM). Embeddings are computed once and cached before any timed run, so this does not affect the reported cold/warm times.

Seeds and randomness. Each seed controls a method's own sources of randomness. Few-Shot, SPILL, and k-LLMmeans query the LLM at temperature 0, where its answers are essentially deterministic, so seeds affect only the k-means initialization and random sampling; Cequel and LSCK-HC ask the LLM five times at temperature 0.7 (randomized answers) and take the majority vote, reproducing the confidence voting of the original papers.

Reproducibility. Because embeddings are precomputed once (Hardware paragraph above), every method clusters identical representations: the design choice that isolates the method's contribution from incidental differences in re-embedding, and that lets future methods slot into the same protocol for a direct, apples-to-apples comparison, addressing the cross-paper comparability problem raised in Section 1. Reproduction scripts and the precomputed embeddings are publicly released, so the numbers reported here can be regenerated without re-querying any API.

Table 2
Datasets (test splits, clustered) and embedding models. K counts the distinct true labels present in the clustered split, N the clustered texts; 59 of MASSIVE's 60 intents occur in its English test split. Access: *open weights* means the model can be downloaded and run locally; *API* means it is available only as an online service.

(a) Datasets

Dataset	K	N	Domain
Banking77 [13]	77	3,080	online banking queries
CLINC150 [14]	150	4,500	task-oriented dialog, out-of-scope class removed
MASSIVE [15]	59	2,974	voice-assistant commands (English)

(b) Embedding models

Model	Dimension	Access
qwen3-embedding-8b [16]	4096	open weights
gemini-embedding-001 [17]	3072	API only
instructor-large [18]	768	open weights

4. Results

4.1. Strength of the Classical Baselines

All values in this section are mean ± standard deviation over 5 seeds; Ward agglomerative is deterministic, hence its ± 0.0. Table 3 establishes the strength of the classical baselines on the same embeddings the LLM methods use. Ward agglomerative is the strongest classical method in 6 of the 9 settings; spectral clustering wins the MASSIVE settings on qwen3 and instructor, and k-means edges out Ward on instructor/Banking77. No single classical method dominates: an LLM method must outperform the best classical method per setting, not just k-means.

Table 3
Classical baseline accuracy (ACC; higher is better; mean ± standard deviation over 5 seeds). **Bold** marks the best classical method per setting. Ward is the strongest classical method in 6 of 9 settings.

Embedding model	Dataset	k-means	Ward	Spectral
gemini-001	Banking77	88.3 ± 0.9	$\mathbf{91.8 \pm 0.0}$	88.0 ± 0.3
gemini-001	CLINC150	85.4 ± 1.3	$\mathbf{89.5 \pm 0.0}$	82.1 ± 0.5
gemini-001	MASSIVE	66.5 ± 2.8	$\mathbf{72.3 \pm 0.0}$	69.8 ± 0.2
qwen3-8b	Banking77	72.6 ± 2.7	$\mathbf{76.3 \pm 0.0}$	74.8 ± 0.4
qwen3-8b	CLINC150	81.9 ± 1.2	$\mathbf{87.6 \pm 0.0}$	77.8 ± 0.1
qwen3-8b	MASSIVE	60.8 ± 3.3	62.0 ± 0.0	$\mathbf{69.1 \pm 1.3}$
instructor	Banking77	$\mathbf{61.6 \pm 2.5}$	61.0 ± 0.0	59.5 ± 0.6
instructor	CLINC150	80.5 ± 1.4	$\mathbf{83.8 \pm 0.0}$	78.9 ± 0.6
instructor	MASSIVE	55.9 ± 2.3	57.8 ± 0.0	$\mathbf{60.0 \pm 1.4}$

4.2. Method Accuracy vs. Baselines

Table 4 shows the core comparison: each table entry stacks the difference in accuracy (ΔACC) vs. k-means (top, the number comparable to the literature) over ΔACC vs. the best classical method (bottom, the more demanding number). The gap between the two lines is the central result: many positive top lines turn negative below, revealing k-means as a weak baseline (Figure 2).

Four findings summarize Table 4:

1. **Exactly one clean win.** Across all 45 method–setting combinations, only SPILL on instructor/Banking77 ($+6.2 \pm 3.8$) outperforms the best classical method by more than one standard

Table 4
LLM method accuracy differences (± standard deviation over 5 seeds). Per entry: top gray line = ΔACC vs. k-means, bottom = ΔACC vs. the best classical method. **Bold** = positive mean vs. the best classical method; † = the only entry exceeding one standard deviation.

Embedding model	Dataset	Few-Shot	k-LLMmeans	SPILL	Cequel	LSCK-HC
gemini-001	Banking77	-2.1 ± 2.1 -5.6 ± 1.3	-2.4 ± 2.6 -5.9 ± 3.2	$+0.5 \pm 1.5$ -3.1 ± 1.4	-11.2 ± 0.6 -14.7 ± 1.1	-14.6 ± 1.4 -18.1 ± 1.8
gemini-001	CLINC150	$+2.0 \pm 1.9$ -2.1 ± 1.3	-0.5 ± 1.2 -4.5 ± 0.7	$+1.8 \pm 1.3$ -2.3 ± 0.4	-6.9 ± 0.3 -11.0 ± 1.0	-3.6 ± 0.4 -7.7 ± 1.2
gemini-001	MASSIVE	$+3.6 \pm 4.6$ -2.2 ± 2.3	$+4.2 \pm 2.1$ -1.6 ± 1.4	$+3.5 \pm 4.8$ -2.4 ± 2.9	-7.8 ± 0.6 -13.6 ± 2.4	-3.1 ± 1.0 -8.9 ± 1.9
qwen3-8b	Banking77	-0.3 ± 2.8 -4.0 ± 0.9	-3.2 ± 3.3 -6.9 ± 1.4	$+3.5 \pm 2.0$ -0.2 ± 2.2	-8.3 ± 1.0 -11.9 ± 2.3	-12.6 ± 3.4 -16.3 ± 4.6
qwen3-8b	CLINC150	$+1.2 \pm 1.6$ -4.5 ± 1.3	-2.1 ± 0.9 -7.8 ± 0.6	$+4.4 \pm 3.0$ -1.3 ± 1.9	-5.9 ± 0.7 -11.6 ± 1.2	-2.6 ± 0.3 -8.3 ± 1.0
qwen3-8b	MASSIVE	$+3.3 \pm 3.0$ -5.1 ± 1.6	$+1.7 \pm 3.4$ -6.7 ± 2.8	$+5.5 \pm 5.1$ -2.9 ± 3.5	-5.9 ± 1.2 -14.2 ± 2.4	-3.3 ± 0.6 -11.7 ± 3.1
instructor	Banking77	$+0.9 \pm 3.2$ $\mathbf{+0.9 \pm 3.2}$	-2.7 ± 2.1 -2.7 ± 2.1	$+6.2 \pm 3.8$ $\mathbf{+6.2 \pm 3.8}^{\dagger}$	-7.4 ± 1.1 -7.4 ± 1.1	-7.8 ± 1.9 -7.8 ± 1.9
instructor	CLINC150	$+1.5 \pm 2.8$ -1.8 ± 1.9	-2.7 ± 2.4 -6.0 ± 2.1	$+3.1 \pm 2.8$ -0.2 ± 1.6	-5.7 ± 0.6 -9.0 ± 1.0	-3.1 ± 0.4 -6.4 ± 1.4
instructor	MASSIVE	$+2.7 \pm 2.4$ -1.4 ± 2.0	$+0.1 \pm 2.0$ -4.0 ± 1.9	$+5.0 \pm 3.6$ $\mathbf{+0.9 \pm 3.0}$	-5.1 ± 0.8 -9.2 ± 1.6	-2.3 ± 0.7 -6.4 ± 1.2

deviation. Two more combinations are positive, but by less than the run-to-run variation (SPILL on instructor/MASSIVE, Few-Shot on instructor/Banking77); every other combination is at or below zero.

2. **Apparent wins against k-means.** SPILL posts a positive ΔACC vs. k-means in all 9 settings ($+0.5$ to $+6.2$), Few-Shot in 7 of 9, and k-LLMmeans in 3 of 9, exactly the comparison the original papers report. Against the best classical method, only SPILL's advantage survives, and only in one setting (Figure 2a).
3. **Interaction with embedding-model strength.** SPILL's ΔACC vs. the best classical method falls steadily as the embedding model gets stronger on Banking77: instructor $+6.2$, qwen3 -0.2, gemini -3.1 (Figure 2b). Neighbor refinement can repair a weak 768-dimensional embedding space but cannot outperform Ward on a strong one (gemini/Banking77 Ward: 91.8).
4. **No method escapes the classical baselines.** The refinement methods land close behind the best classical method (Few-Shot $+0.9$ to -5.6, k-LLMmeans -1.6 to -7.8, SPILL aside); the constraint-based methods trail it in every setting by more (Cequel -7 to -15, LSCK-HC -6 to -18).

4.3. Cost and the Cost–Accuracy Tradeoff

Table 5 reports cold-run cost, in which every LLM response is freshly generated (the stored responses make warm reruns nearly free). The classical baselines cost $0 and finish within 110 seconds; the LLM methods cost $0.002–$0.52 and 1.9–18 minutes cold per dataset run. SPILL is simultaneously the most expensive method and the only one that ever outperforms the best classical baseline. The whole 360-run grid cost $13.95 in billed API spend; cheap in absolute terms, but the ratio is what matters: LLM methods require roughly one hundred to one thousand times more time (for example, SPILL 1,093 seconds vs. Ward 2 seconds) than the classical baselines, in most settings without any accuracy gain.

Cequel's larger call count does not mean larger cost: every call is a uniform 3-utterance, single-letter judgment capped at 4 output tokens, while LSCK-HC's fewer calls are individually longer, each prompt

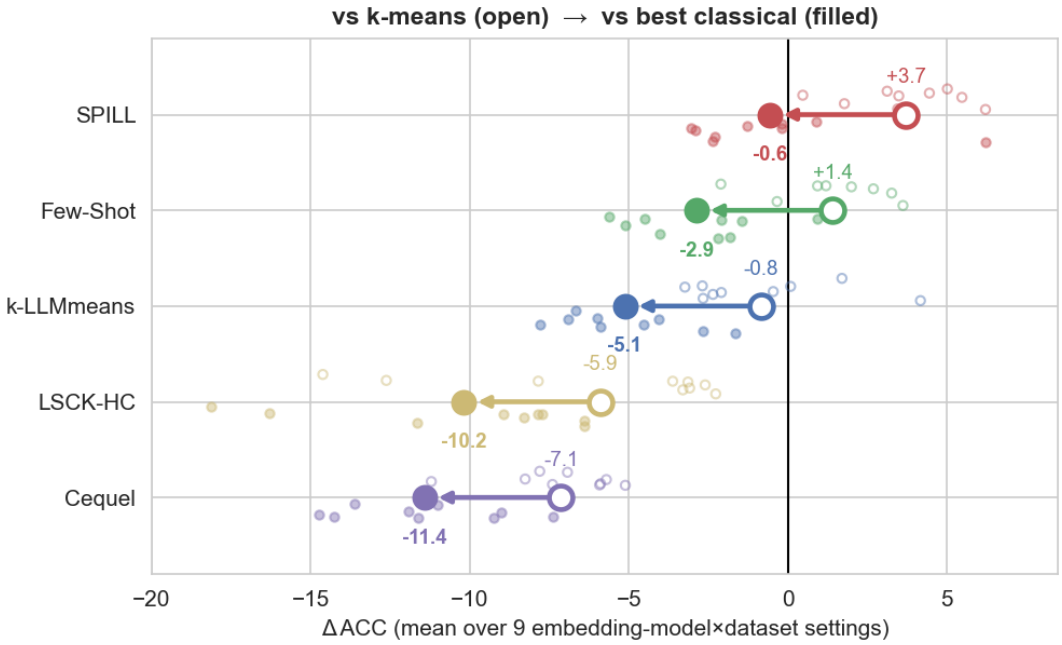


Figure 2: (a) Mean ΔACC per method vs. k-means (open markers) and vs. the best classical method of the setting (filled markers); faint dots = the nine settings. The stronger baseline flips SPILL and Few-Shot from apparent wins to losses. (b) ΔACC vs. the best classical method in each setting, plotted against that setting's best classical accuracy: gains exist only where classical clustering is weak.

Table 5
Cost per run, cold (every LLM query freshly sent); each range spans the nine dataset × embedding-model settings, not a single dataset. LLM calls and US-dollar cost at `deepseek-v4-flash` prices ($0.09 per million input tokens, $0.18 per million output tokens); Cequel and LSCK-HC include the five-fold majority voting. Cold time includes waiting for the API; warm time is computation only, with all LLM responses already stored.

Method	LLM calls	USD	Cold time (seconds)	Warm time (seconds)
k-means	0	$0	6–110	≈ cold
Ward	0	$0	2–46	≈ cold
Spectral	0	$0	2–5	≈ cold
Few-Shot	2,974–4,500 (≈1 per text)	$0.05–0.08	392–675	20–121
k-LLMmeans	71–582	$0.002–0.013	111–418	14–133
SPILL	2,974–4,500 (≈1 per text)	$0.35–0.52	593–1,093	11–53
Cequel	4,685–5,120	$0.10–0.12	351–578	17–107
LSCK-HC	3,072–3,627	$0.11–0.16	405–507	16–97

carrying two worked examples and showing up to 8 texts (must-link) or 6 texts (cannot-link), with output budgets of 128 and 16 tokens; per-call token cost, not call count, sets the total.

Figure 3 synthesizes accuracy and cost. On the strong embedding models, every LLM point sits at or below the best classical point of its setting while costing one hundred to one thousand times more time. On the weak embedding model, only SPILL reaches the slower-but-better region, and convincingly so only on Banking77: is a gain of about 6 accuracy points worth a roughly 100-times longer running time and about $0.40 per run? If a stronger embedding model is available, switching is strictly better: gemini + Ward (91.8) outperforms instructor + SPILL (67.8) by 24 points at $0.

4.4. Reproducibility Gap

Table 6 compares our controlled results to the numbers published in the original papers; "ours" is the mean over the full grid on our best embedding model for that method. Where an exact published

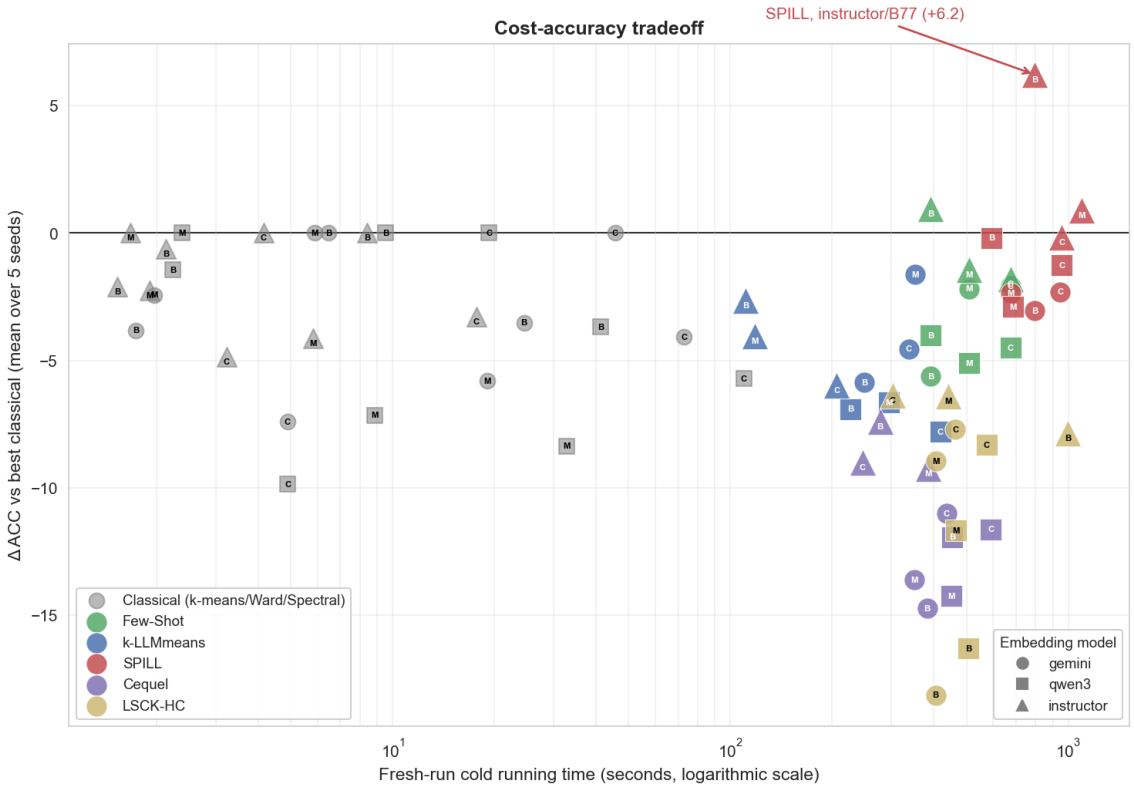


Figure 3: Cost–accuracy tradeoff: one point per (method, embedding model, dataset); marker shape denotes embedding model, and the bold letter inside each marker identifies its dataset (B = Banking77, C = CLINC150, M = MASSIVE); x = cold-run time (logarithmic scale), y = mean ΔACC vs. the best classical method (settings that ran fully from stored responses are estimated from the same method–dataset on another embedding model).

number exists (14 pairs), ours are higher in all 14; LSCK-HC's figure-derived results add four pairs where ours lead on three and tie on CLINC150 NMI. The wins are driven by the embedding model, not by our implementations: on the embedding model the originals actually used (instructor-large), our reproductions land close to or below the published numbers, namely SPILL 67.8 vs. their 71.5 (Banking77) and 83.6 vs. 84.4 (CLINC150); Few-Shot 62.6 vs. 65.3; Cequel 54.2 vs. 69.0 (their GPT-4o-mini and budget of roughly 39,000 LLM queries exceed our `deepseek-v4-flash` at roughly 5,000 queries); LSCK-HC 53.8 vs. 64.4 (theirs uses the stronger DeepSeek-R1); k-LLMmeans on qwen3 matches its published CLINC150 accuracy (79.8 vs. 80.2). The reproducibility gap in absolute numbers is mostly an embedding-model gap; the relative finding (LLM methods vs. the classical baselines) is consistent across all three embedding models. No published numbers exist for MASSIVE under our protocol for any method, hence the table covers Banking77 and CLINC150 only.

5. Discussion

5.1. When Do LLM Methods Win?

Our results narrow "when do LLM-guided clustering methods outperform classical baselines" to one specific condition: a weak embedding model, paired with neighbor-selection refinement, on a fine-grained task. The one surviving win (SPILL on instructor/Banking77) loses its edge on stronger embedding models (qwen3 -0.2, gemini -3.1): the advantage exists only where the embedding space fails to separate the intents on its own, leaving structure that the LLM signal can still recover; stronger embedding models capture this structure themselves.

This pattern is consistent with what SPILL does: averaging each text's embedding with those of its LLM-selected same-intent neighbors is a form of local smoothing, which helps most when the repre-

Table 6
Our results vs. published numbers (ACC/NMI). "Ours" uses our best embedding model for the method (in parentheses); **bold** marks the better side of each comparable pair. LSCK-HC reports these datasets only graphically: Banking77 ACC is printed in their Figure 2 (DeepSeek R1, 30% constraints); ≈ values are read from the plateaus of their Figure 1. Cequel's paper has no CLINC150 intent-discovery result (its "CLINC" task clusters the same texts into 10 broad domains). Absolute gaps are mostly embedding-model gaps; see text.

	Method	Ours (embedding model)	ACC/NMI	Published	Their embedding model	Their LLM
Banking77	Few-Shot	gemini	**86.2/92.8**	65.3/82.4	Instructor	GPT-3.5
	k-LLMmeans	gemini	**85.9/93.1**	67.9/84.1	text-emb-3-small	GPT-4o
	SPILL	gemini	**88.8/93.8**	71.5/85.1	Instructor	Qwen (local)
	Cequel	gemini	**77.1/85.0**	69.0/83.2	Instructor	GPT-4o-mini
	LSCK-HC	gemini	**73.7/84.6**	64.4/≈82	Instructor	DeepSeek R1/V3
CLINC150	Few-Shot	gemini	**87.4/95.8**	79.4/92.6	Instructor	GPT-3.5
	k-LLMmeans	gemini	**85.0/95.2**	80.2/93.1	text-emb-3-small	GPT-4o
	SPILL	gemini	**87.2/95.8**	84.4/93.6	Instructor	Qwen (local)
	Cequel	gemini	78.5/90.0	—	—	—
	LSCK-HC	gemini	81.8/92.5	≈79/≈93	Instructor	DeepSeek R1/V3

sentation is noisy (instructor-large: 768-dimensional, general-purpose) and helps least (or even hurts) when the intents are already well separated. The other methods show no such compensating behavior: their accuracy differences vs. the best classical method are negative in every setting, and Figure 2b shows all methods trending downward as the classical baseline strengthens. An LLM refinement signal is only informative relative to what the embedding model already captures, so the practical question is whether the embedding model is weak enough to benefit from LLM-guided clustering.

5.2. Is the Cost Justified?

SPILL is the most expensive method in our benchmark and the only one that ever yields a net gain; Section 4.3 makes the alternative explicit: switching from instructor + SPILL (67.8 ACC, about $0.40, about 18 minutes) to gemini + Ward (91.8 ACC, $0, seconds) is better on every axis. The relevant comparison is therefore an LLM method against spending the same effort on a better embedding model, and in every case we tested, the better-embedding-model option wins. The additional cost is justified only where upgrading the embedding model is not an option; even then, the expected gain is a few accuracy points on fine-grained tasks.

5.3. Limitations

Single LLM. All methods share `deepseek-v4-flash`. Stronger or weaker LLMs may shift absolute results, though we expect the relative finding (gains shrink as embedding quality rises) to hold, since it is a property of the classical baseline's strength, not of the LLM.

Implementation fidelity. We reimplemented the five methods from their papers rather than running original author code (mostly unreleased or partial). Implementation choices such as prompt wording or output length limits can materially affect a method's measured accuracy; we fixed the issues we found but did not systematically try all implementation variants.

One task family. All three datasets are short-text intent detection with a known number of clusters K. Results may not transfer to long-document clustering, tasks where K is not known in advance, or domains where strong embeddings are unavailable (low-resource languages, specialized jargon), where LLM refinement might plausibly add more value.

Cost model. Dollar and time figures reflect one provider's pricing and response times at one point in time; falling LLM prices could shift the tradeoff.

Five methods, one pipeline shape. We selected five representative embedding-refinement methods (Section 2). Other classes of LLM-based clustering methods (those that construct labeled cluster hierarchies, those that retrain the embedding model on LLM-generated supervision, and those that process documents as they arrive) were out of scope and may show different cost–accuracy profiles.

6. Conclusion

Do LLM-guided clustering methods outperform classical clustering baselines, and at what cost? Across 360 controlled runs, the answer is mostly no. Against k-means, the comparison the original papers report, the refinement methods usually outperform it (SPILL in all nine settings); the constraint-based methods never do. Against the best classical baseline per setting (usually Ward agglomerative), the advantage almost entirely disappears: exactly one of the 45 method–setting combinations (SPILL on instructor/Banking77, $+6.2 \pm 3.8$ ACC) outperforms classical clustering by more than its variation across seeds, at one hundred to one thousand times the cost of the classical baselines, which require no LLM queries at all.

The apparent gains in the literature largely reflect a weak baseline. Our reproductions are frequently higher than the originals (Section 4.4), so published numbers are not the problem. The comparison itself, LLM-refined vs. k-means, understates how strong classical clustering is. Once agglomerative and spectral baselines run on the same embeddings alongside k-means, the value of LLM refinement collapses to a narrow set of conditions: weak embedding models, fine-grained class structure, neighbor-smoothing refinement. Even there, the same gain is available for free by switching to a stronger embedding model.

Our contribution is less the specific answer than the protocol behind it: same datasets, same embedding models, same LLM, multiple seeds, and classical baselines strong enough to make the comparison meaningful. We release the benchmark, the precomputed embeddings, and the cost-tracking tooling so that future methods can be evaluated against this bar rather than against k-means alone. In future work, we plan to extend the protocol to other pipeline shapes and to domains where strong embeddings are genuinely unavailable. Until then, the evidence supports a plain recommendation: before reaching for an LLM-guided clustering method, try a better embedding model and Ward agglomerative clustering first.

Acknowledgments

This article was produced with the financial support of the project: Biography of Fake News with a Touch of AI: Dangerous Phenomenon through the Prism of Modern Human Sciences, project no. CZ.02.01.01/00/23_025/0008724, via the Operational Programme Jan Ámos Komenský. The required compute was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

Declaration on Generative AI

During the preparation of this work, the author used Claude (Anthropic) to assist with coding, data analysis, and writing. After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] S. P. Lloyd, Least squares quantization in PCM, IEEE Trans. Inf. Theory 28 (1982) 129–137. doi:10.1109/TIT.1982.1056489.

[2] J. H. Ward, Hierarchical grouping to optimize an objective function, J. Am. Stat. Assoc. 58 (1963) 236–244. doi:10.1080/01621459.1963.10500845.
[3] U. von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416. doi:10.1007/s11222-007-9033-z.
[4] Y. Zhang, Z. Wang, J. Shang, ClusterLLM: Large language models as a guide for text clustering, in: Proceedings of EMNLP 2023, 2023, pp. 13903–13920. doi:10.18653/v1/2023.emnlp-main.858.
[5] V. Viswanathan, K. Gashteovski, C. Lawrence, T. Wu, G. Neubig, Large language models enable few-shot clustering, Trans. Assoc. Comput. Linguist. 12 (2024) 321–333. doi:10.1162/tacl_a_00648.
[6] J. Diaz-Rodriguez, Summaries as centroids for interpretable and scalable text clustering, in: Proceedings of ICLR 2026, 2026. ArXiv:2502.09667.
[7] I.-F. Lin, F. Hasibi, S. Verberne, SPILL: Domain-adaptive intent clustering based on selection and pooling with large language models, 2025. arXiv:2503.15351.
[8] H. Wang, T. Zhang, R. Yang, J. Xu, Cequel: Cost-effective querying of large language models for text clustering, 2025. arXiv:2504.15640.
[9] C. Jia, W. Wu, L. Guo, Z. Lu, C. Chen, K.-L. Ong, Optimized algorithms for text clustering with LLM-generated constraints, in: Proceedings of AAAI 2026, 2026, pp. 22229–22237. doi:10.1609/aaai.v40i27.39379.
[10] DeepSeek, DeepSeek-V4-Flash model card, https://openrouter.ai/deepseek/deepseek-v4-flash, 2026. Accessed July 2026.
[11] F. Pedregosa, G. Varoquaux, A. Gramfort, et al., Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[12] H. W. Kuhn, The hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1955) 83–97. doi:10.1002/nav.3800020109.
[13] I. Casanueva, T. Temčinas, D. Gerz, M. Henderson, I. Vulić, Efficient intent detection with dual sentence encoders, in: Proceedings of the 2nd Workshop on NLP for Conversational AI, 2020, pp. 38–45. doi:10.18653/v1/2020.nlp4convai-1.5.
[14] S. Larson, A. Mahendran, J. J. Peper, et al., An evaluation dataset for intent classification and out-of-scope prediction, in: Proceedings of EMNLP-IJCNLP 2019, 2019, pp. 1311–1316. doi:10.18653/v1/D19-1131.
[15] J. FitzGerald, C. Hench, C. Peris, et al., MASSIVE: A 1M-example multilingual natural language understanding dataset with 51 typologically-diverse languages, in: Proceedings of ACL 2023 (Volume 1: Long Papers), 2023, pp. 4277–4302. doi:10.18653/v1/2023.acl-long.235.
[16] Y. Zhang, M. Li, D. Long, et al., Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. arXiv:2506.05176.
[17] J. Lee, Z. Dai, S. M. K. Duddu, T. Lei, I. Naim, M.-W. Chang, V. Y. Zhao, et al., Gemini embedding: Generalizable embeddings from Gemini, 2025. arXiv:2503.07891.
[18] H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W.-t. Yih, N. A. Smith, L. Zettlemoyer, T. Yu, One embedder, any task: Instruction-finetuned text embeddings, in: Findings of ACL 2023, 2023, pp. 1102–1121. doi:10.18653/v1/2023.findings-acl.71.