SPILL: Zero-shot Intent Clustering based on Selection and Pooling with Large Language Models

Anonymous ACL submission

Abstract

In this paper, we propose Selection and Pooling with Large Language Models (SPILL), an intuitive, zero-shot method for intent clustering without fine-tuning. Existing embeddings-based clustering methods rely on a few labeled examples or unsupervised fine-tuning to optimize results for each new dataset, which makes them less generalizable to multiple datasets. Our goal is to make these existing embedders more generalizable to new domain datasets without further fine-tuning. Inspired by our theoretical derivation and simulation results on the effectiveness of sampling and pooling techniques, we view the clustering task as a small-scale selection problem. A good solution to this problem is associated with better clustering performance. Accordingly, we propose a two-stage approach: First, for each utterance (referred to as the seed), we derive its embedding using an existing embedder. Then, we apply a distance metric to select a pool of candidates close to the seed. Because the embedder is not optimized for new datasets, in the second stage, we use an LLM to further select utterances from these candidates that share the same intent as the seed. Finally, we pool these selected candidates with the seed to derive a refined embedding for the seed. We found that our method generally outperforms directly using an embedder, and it achieves comparable results to other state-of-the-art studies, even those that use much larger models and require fine-tuning, showing its strength and efficiency. Our results indicate that our method enables existing embedders to be further improved without additional fine-tuning, making them more adaptable to new domain datasets. Additionally, viewing the clustering task as a small-scale selection problem gives the potential of using LLMs to customize clustering tasks according to the user's goals.[1]

[1]The source code is available: https://anonymous.4open.science/r/paper_review-5168/README.md

1 Introduction

Intent detection is a fundamental component in task-oriented dialogue (TOD) systems, aimed at classifying user utterances into pre-defined intent categories (Ni et al., 2023). Although some research has focused on addressing data scarcity (Siddique et al., 2021; Lin et al., 2024), pre-defined intent labels are insufficient for addressing all user requests, as new intents emerge with growing complexity of customer requirements and appearance of novel domains on the business front. While the progress of transformer-based models has greatly enhanced intent detection performance, the identification of emerging intents in task-based conversational agents continues to present a challenge (Zhou et al., 2023; Rodriguez et al., 2024).

To address this issue, the majority of research aims to develop embedding models that group unlabeled utterances into clusters based on a labeled or unlabeled in-domain dataset (Zhang et al., 2021b; Mou et al., 2022a; Zhang et al., 2023a; Liang and Liao, 2023). The goal of these approaches is to enable the embedder to learn a robust representation of user utterances while aligning with the cluster objective. Contrastive learning is commonly employed for this purpose, aiming to learn a representation through comparison (Le-Khac et al., 2020). The clustering objective is achieved through the design of a cluster loss function (Zhang et al., 2021a; Mou et al., 2022b; Du et al., 2023). Although these approaches yield good results, they require fine-tuning for each dataset. (Zhang et al., 2021a).

In recent years, advancements in generative large language models (LLMs) (Touvron et al., 2023; Team, 2024a) have facilitated improvements in intent clustering. Zhang et al. (2023b) and Liang et al. (2024) used LLMs to guide the fine-tuning of embedders, aiming to align the embeddings' clustering outcomes with LLMs' interpretations. Although these studies achieve state-of-the-art results,

they face two primary challenges: First, modifying the loss function adds complexity, as it involves designing different loss functions and tuning additional hyperparameters, such as the weight of each loss term. This makes optimization more difficult. Second, building a new embedder requires optimization for each dataset, which limits generalizability.

In this paper, we propose a theoretical framework for clustering, grounded by formal proofs, and an intuitive and effective approach to address these challenges. Our approach has three key goals: easy implementation, no need for fine-tuning, and the ability to adapt to unseen datasets. Our approach stays close to a theoretical rationale and we confirm its potential by simulation analysis. The key idea is that if we can identify a few utterances that share the same cluster as the seed utterance from a randomly selected subset, pooling the seed utterance with these selected utterances will bring them closer to the cluster centroid. Based on this premise, we can see a clustering task as a small-scale selection problems.

Our approach consists of two stages: In the first stage, for each utterance (referred to as the "seed utterance"), we use an existing embedder (a traditional encoder or a decoder-only LLM) to gain a larger pool of similar utterances. In the second stage, we use LLMs to further select utterances that share the same intent cluster as the seed utterance.

Note that our proposed approach is not intended to compete with other embedders. Instead, it is designed to complement most existing approaches and can strengthen each of the existing models. With our experiments, we show that our method can boost performance on the clustering task irrespective of the used embedder. In summary, we make four contributions: (1) we provide a theoretical framework supported by formal proofs and simulations, which frame the clustering task as a small-scale selection problem, providing both theoretical and empirical contributions to the task. (2) We propose a novel and easy-to-implement approach that is generalizable, regardless of the embedders used and does not require fine-tuning and can operate with low computational resources; (3) We show our method enables domain adaptation in clustering for unseen datasets, achieving state-of-the-art results on four benchmark collections.

2 Related Work

Intent clustering with contrastive learning Grouping user utterances and identifying new intents is essential in TOD systems (Zhang et al., 2021a,b; Mou et al., 2022a; Liang and Liao, 2023; Du et al., 2023). Most research has focused on developing embedding models to create strong representations of user utterances. For instance, Zhang et al. (2021b) pretrained a model with little labelled data and use k-means to produce cluster assignments as pseudo labels. They learn the intent representations under the supervision of the aligned pseudo-labels. Zhang et al. (2021a) propose a method that optimizes both the contrastive loss and clustering loss together to build a sentence embedding model. To prevent overfitting on in-domain data during contrastive loss optimization, Mou et al. (2022a) limit the comparison to k-nearest neighbors instead of considering all possible neighbors. As earlier research focused on contrastive learning without fully accounting for the semantic meanings of labels, Liang and Liao (2023) use two-level contrastive learning to learn representations. This approach first aligns embeddings with several contrastive objectives, including their proposed label semantic alignment, then applies soft prompting to enhance the use of semantic knowledge in intent discrimination.

Sentence embeddings with LLM feedback With the advancement of LLMs, research is increasingly using their capabilities to improve embedding. Zhang et al. (2023b) introduce a method that constructs multiple triplet questions, each consisting of an anchor data point and two candidate points. The triplets are initially selected from different clusters using a smaller embedder, and the LLM is then tasked with identifying the positive pair for the anchor point. After fine-tuning the embedder, they perform an initial clustering using the updated embeddings and then leverage the LLM to refine the clustering granularity. Their results show substantially better performance than traditional embedding methods like **SCCL** (Zhang et al., 2021a) while using less data for training. Liang et al. (2024) leverage LLMs to derive intent descriptors. They then design contrastive loss functions to optimize a smaller embedder, synergizing LLMs and smaller language models for intent recognition. Instead of fine-tuning an embedder, Viswanathan et al. (2024) use LLMs to improve the utterances

by having them generate key phrases for each sentence, which are then added to the sentence and encoded into embeddings. De Raedt et al. (2023) select prototypical utterances, generate labels for non-prototypical ones using LLMs, and encode both utterances and labels together.

Sentence embeddings in LLMs While previous research has focused on methods that rely on pre-trained contrastive loss embedders with feedback from LLMs, recent studies have shown that directly extracting embeddings from LLMs can also be effective. Jiang et al. (2023) propose an in-context learning approach to improve embeddings by introducing a "one word limitation". The idea is to instruct LLMs to summarize the input sentence into a single word. They found that this approach can still achieve good performance without fine-tuning. Springer et al. (2024) propose the echo embedding approach. Because auto-regressive embeddings do not capture context from later tokens, they pass the sentence through the model twice and pool embeddings only from the second occurrence. Their experiment shows that this method outperforms traditional pooling. Lei et al. (2024) propose a meta-task prompting method with a 'one-word limitation.' The embeddings are created using a series of carefully designed prompts that cover different aspects of meaning, using the 'one-word limitation' to improve the embeddings.

3 Theoretical framework

In this section, we introduce our theoretical derivation, followed by validations through empirical simulations.

3.1 Problem formulation

We cast the problem of identifying emerging intents to a clustering task, where conversation utterances are grouped into clusters, with each cluster corresponding to a newly identified intent. Consider a collection of data points $D = \{x_i\}_{i=1}^{N}$, where each data point $x_i \in D$ corresponds to an utterance, and N is the total number of data points. The task is to partition D into cluster sets $\{\hat{S}_l\}_{l=1}^{\hat{M}}$, where $\hat{M}$ is the number of the unique clusters. Note that $\sum_{l=1}^{\hat{M}} |\hat{S}_l| = N$. The objective is to ensure that the clustering results $\{\hat{S}_l\}_{l=1}^{\hat{M}}$ are as close as possible to the true partition $\{S_l\}_{l=1}^{M}$, where M is the number of unique clusters in the ground truth. Similarly, $\sum_{l=1}^{M} |S_l| = N$.

3.2 Theoretical grounding and proof

For simplicity in the theoretical development, we assume sampling is performed with replacement. Consider cluster $S \in \{S_l\}_{l=1}^{M}$, containing N_S data points such that $S = \{x_i\}_{i=1}^{N_S}$. For each data point x_i, we derive a d dimensional vector representation $\mathbf{z}_i \in \mathbb{R}^d$ from an embedder. This results in the set $\{\mathbf{z}_i\}_{i=1}^{N_S}$. We denote σ_h^2 and μ_h as the variance and mean of the cluster for the h-th dimension of our d-dimensional space. μ_h is also the cluster centroid in h-th dimension. If we randomly select one utterance from the cluster S, denoted as $\mathbf{Z}_i$,[2] then we randomly select k elements with replacement $\mathbf{Z}_{i1}, \mathbf{Z}_{i2}, ..., \mathbf{Z}_{ik}$ from $\{\mathbf{z}_i\}_{i=1}^{N_S}$ to derive its pooling version $\overset{*}{\mathbf{Z}}_i$, defined as follow:

$$\overset{*}{\mathbf{Z}}_i := \frac{\mathbf{Z}_i + \sum_{m=1}^{k} \mathbf{Z}_{im}}{1+k} \quad (1)$$

Based on this, we formally prove the following inequality holds (a detailed proof is provided in Appendix A):

$$\sum_{h=1}^{d} E[(\overset{*}{Z}_{ih} - \mu_h)^2] - E[(Z_{ih} - \mu_h)^2] < 0, \quad (2)$$

where $Z_{ih} \in R$ and $\overset{*}{Z}_{ih} \in R$ represent elemnts of the h-th dimension of $\mathbf{Z}_i$ and $\overset{*}{\mathbf{Z}}_i$, respectively.

This inequality suggests that using this sampling and pooling approach, the samples are closer to the cluster centroid, which implies that clustering can be approached as a small-scale selection task. Specifically, instead of identifying all similar utterances at once, we only need to determine whether a randomly chosen subset of utterances belongs to the same cluster as the seed utterance. The idea based on the inequality is that if we can perform selection perfectly on the subset of data points, then any cluster will show reduced variance when each data point is pooled with randomly selected points from the same cluster, compared to the variation in the original points. We examine the relationship between variance and clustering performance in Section 3.3 using simulations.

3.3 Simulation study

Inspired by the theoretical principles outlined above, our main goal is to identify utterances that share the same intent as the seed utterance. To

[2]We use capital to highlight it is a random vector

achieve this, we only need to consider a randomly selected subset of candidate utterances at a time.

A straightforward way to compare the seed utterance with a subset is to use an LLM directly. However, choosing the right subset size is challenging: A small subset may miss related utterances, while a large one increases computational cost. To address this, we use an embedding model to select a subset of similar utterances as a starting point. However, this approach might deviate from the theory's assumption that the subset is randomly selected from the whole cluster, as it focuses only on locally close utterances. To evaluate the impact brought by embedder-base selection, we conduct a simulation study.

In the simulation, we analyze two extreme pooling strategies: randomly selecting k utterances from the same cluster to pool with the seed utterance (strategy **Rd**) (following the theoretical analysis), and selecting only the top-k closest utterances from the same cluster to pool with the seed utterance (strategy **TopK**). Sampling is performed without replacement to prevent over-representation of certain examples, which can occur in small clusters. See Appendix B for simulation process details.

Table 1 shows that both **Rd** and **TopK** pooling strategies reduce variances while simultaneously improving clustering metrics in both non-skewed and skewed settings. Furthermore, the results indicate that increasing the pooling size leads to better clustering performance. Finally, even in scenarios with a skewed distribution, both pooling methods continue to enhance performance. We also investigate whether the results hold with varying dimensionality. We found that for higher dimensionality, the variances for both **Rd** and **TopK** remain reduced and the clustering performance improved (more details in Appendix C).

The findings of our simulation study are as follows: (1) lower variance within a cluster is associated with better clustering performance. (2) increasing the value of k tends to improve clustering performance. (3) pooling based on the closest k still shows improvement, indicating that using an embedder as an initial selection step is a reasonable approach.

	k = 0	k = 5		k = 10		k = 20	
		Rd	TopK	Rd	TopK	Rd	TopK
Normal							
$\hat{var}^2$	39.76	6.33	7.70	3.30	4.20	1.58	2.12
Acc	37.54	69.73	65.74	94.71	92.61	99.54	99.20
NMI	0.52	33.15	32.94	80.32	77.32	97.60	96.33
Log-normal							
$\hat{var}^2$	28.12	4.45	3.04	2.33	1.29	1.12	0.47
Acc	43.52	47.54	53.50	58.60	88.16	81.67	98.77
NMI	0.93	9.82	19.14	31.63	71.41	74.94	94.16

Table 1: Average estimated variances, mean, and clustering metrics over 50 runs in our simulation study. Note: $\hat{var}^2$: Estimated variance for the one of the cluster (We report only one of them since all clusters have the same trend) over 128 dimensions; k: Number of data points selected to pool with the seed.

4 Computational Method

Based on our theoretical analysis and simulation results, randomly selecting utterances among the entire cluster to pool with a seed utterance is guaranteed to reduce clustering variance, which is associated with better clustering performance. The simulation results further demonstrate that pooling using top$-k$ selection can still outperform no pooling and can approach the performance of random selection achieved with larger k in higher dimensional spaces. Building on these findings, we propose a two-stage selection approach. In the first stage, we implement a selection strategy based on these findings. In the second stage, to refine the candidates from the first stage, we leverage the capabilities of LLM to identify the best utterances in the selection, ensuring that the selected utterances belong to the same cluster.

4.1 First stage: Embedding based selection

In this stage, we use open-source models to extract the utterance embeddings: We feed x_i into a existing pretrained encoder or a large language model (referred to as the "embedder" in the following text) to extract its embedding, denoted as $\mathbf{z}_i := \text{Embedder}(x_i)$. Let $d(\mathbf{z}_i, \mathbf{z}_j)$ be a distance function that compute the distance between the two embeddings $\mathbf{z}_i$ and $\mathbf{z}_j$, where $i \neq j$. This distance is denoted as d_{ij}. We use Euclidean distance. By sorting the distances d_{ij}, we can select the top l_{top} closest utterances to the seed x_i, which have highest chance of being in the same cluster. These corresponding utterances are denoted as $\{x_{ij}\}_{j=1}^{l_{top}}$. Additionally, to introduce some variety, we also select utterances, denoted as l_{random} $\{x_{ij}\}_{j=1}^{l_{random}}$, using chunk sampling: We split the whole sample into l_{random} chunks, and take the first closest utterance from each chunk. This chunk sampling method introduces variety while ensuring that relatively close utterances are selected. Thus, the entire candidate set is given by: $D_i := \{x_{ij}\}_{j=1}^{l_{top}} \cup$

$\{x_{ij}\}_{j=1}^{l_{random}}$ The goal of the first stage is to quickly select utterances that are more likely to belong to the same cluster as the seed x_i. However, since the embedder is not specifically optimized for the datasets we are experimenting with, and the clusters may be in overlapping areas in the vector space, it is possible that some candidates belong to a different cluster than the seed x_i.

4.2 Second stage: LLM based selection

To address the possibility that the candidate set D_i may include utterances from different clusters than x_i, we use in-context learning with an LLM to select utterances that share the same intent as the seed utterance from D_i. Note that D_i is shuffled before being fed into the LLM. These selected utterances are denoted as $\tilde{D}_i$, note $\tilde{D}_i \subseteq D_i$. We then take the seed x_i along with all elements from $\tilde{D}i$ to compute the mean pooling. This pooled representation, $\overset{*}{\mathbf{z}}_i$, will be used for the clustering algorithm. The size of $\tilde{D}_i$ varies for each seed utterance, as it depends on the LLM's selection result.

The purpose for the second stage is to leverage the power of LLMs with a simple designed prompt to further select the utterances sharing the same intent as the seed utterance.

5 Experimental Settings

5.1 Datasets and models

We use **SGD** (Rastogi et al., 2020), **Bank77** (Casanueva et al., 2020), **CLINC150** (Larson et al., 2019), **Mtop** and **Massive** for our experiments. Statistics are provided in Table 2.

In the first stage, embeddings extraction, we experiment with three decoder models, i.e. **Qwen2.5-7B-Instruct** (Team, 2024b) **Llama3.2-8B-Instruct** (AI@Meta, 2024), and **Gemma-2-9b-it** (Team, 2024a), as well as two encoder models, i.e. **E5-large** (Wang et al., 2022) and **Instructor-large** (Su et al., 2022). In the second stage, we use an LLM to verify these utterances. If a decoder model is used in the first stage as an embedder, we continue with the same model in the second stage; for the encoder models in the first stage, we tried all the three LLMs in the second stage. For simplicity, we refer to these models with shortened names: Qwen, Llama, Gemma, E5, and Instructor in the following sections.

Dataset	# clusters	# utterances
SGD	34	$1,506^*$
Bank77	77	3,080
Clinc150	150	4,500
Mtop	102	4,386
Massive	59	2,974

Table 2: Dataset Statistics. Note: **SGD** includes a dataset of 57.2K samples across the training, validation, and test splits. We only use its subset for setting hyperparameters. For other datasets, we use the same settings as Zhang et al. (2023b)

Task Instructions:

Step 1: Identify Intent Clusters
Review the Candidate Utterances to identify their individual intents and group them into clusters based on shared intent. Candidates may either align with the same cluster as the Target Utterance or belong to entirely different clusters.
Note: Intent refers to the request or the purpose the user wants to achieve.

Step 2: Match Intent with Target Utterance
Compare each Candidate's intent to the Target Utterance, using the clusters you identified. Select only Candidates from the same intent cluster as the Target Utterance.
Note: Choose a Candidate only if its intent clearly aligns with the Target Utterance's purpose.

Answer Format:
Only provide the final selection of Candidate Utterances by listing their numbers if they match the Target Utterance intent or request.
1. If Candidates 3, 4, 9, and 11 match, write: The Candidate utterances numbers are: 3, 4, 9, 11
2. If no Candidate matches, write: The Candidate utterances numbers are: none

Note: Stick to the answer format and avoid providing extra explanations.

Task:
Target Utterance: {sentence 1}
Candidate Utterances:
1. {sentence 1}
...
L. {sentence L}

Table 3: Task Instructions Prompt. Note: The boldface used here is for readability; it is not used in the prompt.

5.2 Prompts used

First stage We build on previous research to implement our method for extracting embeddings. The prompts we used can be see in appendix D. We in general follow their practice.

Second stage In this stage we use the same prompt across different LLMs, shown in Table 3.

5.3 Embedding derivation

E5 and **Instructor** use mean pooling to derive embeddings for sentence representation. **Echo** uses mean pooling on the last hidden layer output to achieve better results than using the last token, while **Summarizer** utilizes the last token embedding from the last hidden layer output. We adopt their settings for our follow-up experiments.

5.4 Evaluation

To evaluate the embedding quality, for simplicity, we assume the true cluster number is known

(note that in our method, this is not required), in line with previous research evaluation practices (Zhang et al., 2021a, 2022, 2023b; Viswanathan et al., 2024; Liang et al., 2024). Therefore, the number of clusters will correspond to the number of labels for each dataset, and the KMeans algorithm (Lloyd, 1982) is applied for clustering.

Since the datasets used in our experiments are labeled, we will apply standard clustering metrics to evaluate the results. These metrics include normalized mutual information (NMI) and clustering accuracy (Acc) (Rand, 1971; Meilă, 2007; Huang et al., 2014; Gung et al., 2023).

5.5 Hyperparameters setting

Although our proposed method does not require updating the model parameters, it relies on one key hyperparameter: the values used to select first-stage candidates, specifically the values of l_{top} and l_{random}. We use **SGD** as an external dataset to optimize these parameters. This avoids directly applying our method to the evaluation datasets, ensuring a fair comparison. We set our *tolerance*[3]: at $20 = l_{top} + l_{random}$, and we experiment with different combinations of $l_{top} \in \{2, 4, ..., 20\}$. We found that the combination $l_{top} = 14; l_{random} = 6$ gives the best performance, though overall results show little difference across combinations. (See more detail in appendix E). We fix the setting (14,6) throughout our experiments across all different datasets and models.

5.6 Baselines

We compare our method with the state-of-the-art (SOTA) approaches, all of which operate in a zero-shot setting (i.e., performing tasks on an unlabeled dataset). Among these, **SynCID** and **ClusterLLM** involve fine-tuning of their embedding models.

SynCID (Liang et al., 2024) use **davinci-003** to refine utterances, which are then used to fine-tune a small embedder, **bert-base-uncased**.[4] We report their zero-shot setting result.

KeyphraseCluster (Viswanathan et al., 2024) uses **GPT** (gpt-3.5-turbo) to generate key phrases for each utterance, which are then concatenated for clustering. They use **Instructor-large** as their backbone of the embedder (Su et al., 2022).

[3]we call tolerance because increasing it adds more computation in the second stage

[4]google-bert/bert-base-uncased

ClusterLLM (Zhang et al., 2023b) uses **GPT** (gpt-3.5-turbo) to construct hard triplets from the test dataset and fine-tune a small embedder. They use **Instructor-large** (Su et al., 2022) and **E5-large** embedders (Wang et al., 2022).

Echo embeddings and Summarizer Both Echo embeddings (Springer et al., 2024) and Summarizer (Lei et al., 2024) do not rely on contrastive learning for optimization. Instead, they use their own prompts to derive the desired embedding from decoder-only LLMs.

6 Results

6.1 Main results

Tabel 4 compares our proposed method SPILL with all baselines with different encoder and decoder embedders. For encoder embedders, we found that our proposed method generally performs best with **Gemma**, followed by **Qwen**, and then **LLama**.

Additionally, SPILL mostly in almost all setting performs better than directly using the embeddings. Notably, our proposed method achieves results comparable to Zhang et al. (2023b) (ClusterLLM) and Liang et al. (2024) (SynCID), which perform fine-tuning on the embedders. For decoder embedders, our method shows better performance than **Echo** and **Summarizer** on most datasets, and even better than encoder embedders on some datasets. This suggests that decoder embedders without contrastive loss optimization can outperform encoder embedders and show self-improvement with SPILL.

6.2 Ablation and analysis

Effectiveness of the first and second stage We analyze how the first and second stages improve clustering performance. Based on the results shown in Table 5, we find that the first selection stage generally improves performance compared to the plain setting, and the second selection stage often provides further performance gains compared to the first stage.

Analysis of Performance Variations Although Table 4 shows that our proposed method mostly improves performance than directly using an embedder, we analyze why some results still show lower performance. We found a higher correct selection ratio is associated with better clustering performance (See more detail in appendix F). We

	Bank 77		Clinc150		Mtop		Massive	
	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc
Fine-tuned embeddings								
SynCID (davinci-003)	84.20	**72.89**	94.23	86.80	-	-	-	-
ClusterLLM$_{E5}$ (GPT)	84.16 (0.36)	70.13 (1.34)	92.92 (0.29)	80.48 (0.93)	74.46 (0.11)	37.22 (1.18)	74.39 (0.21)	56.08 (1.01)
ClusterLLM$_{Instructor}$ (GPT)	**85.15** (0.41)	71.20 (1.59)	94.00 (0.21)	83.80 (0.41)	73.83 (0.79)	35.04 (0.97)	77.64 (0.21)	60.69 (0.96)
E5-large								
Plain	77.19 (0.34)	59.60 (1.42)	91.27 (0.38)	75.92 (0.91)	70.87 (0.23)	34.21 (0.57)	71.38 (0.55)	53.85 (1.28)
SPILL (Gemma)	83.56 (0.47)	70.25 (1.56)	92.93 (0.08)	83.18 (0.78)	71.77 (0.35)	36.83 (1.10)	75.40 (0.51)	60.28 (1.81)
SPILL (LLama)	80.94 (0.29)	66.84 (0.97)	91.41 (0.07)	80.09 (1.31)	70.52 (0.51)	35.04 (1.06)	72.74 (0.44)	58.06 (1.52)
SPILL (Qwen)	83.64 (0.26)	70.31 (0.75)	92.67 (0.31)	82.58 (0.87)	71.20 (0.36)	36.48 (0.73)	74.46 (0.57)	59.38 (1.20)
Instructor-large								
Plain	82.38 (0.59)	65.70 (1.78)	93.25 (0.32)	81.12 (2.27)	71.69 (0.60)	34.06 (2.51)	74.56 (0.37)	56.62 (1.75)
KeyphraseClust. (GPT)	82.4 (0.0)	65.3 (0.0)	92.6 (0.0)	79.4 (0.0)	-	-	-	-
SPILL (Gemma)	85.01 (0.29)	71.05 (0.83)	93.77 (0.35)	85.14 (1.04)	72.65 (0.32)	37.11 (1.62)	77.62 (0.46)	62.42 (2.06)
SPILL (LLama)	83.37 (0.22)	69.55 (0.60)	92.96 (0.18)	84.31 (0.72)	71.41 (0.31)	35.18 (1.12)	75.28 (0.83)	58.79 (2.75)
SPILL (Qwen)	85.12 (0.30)	71.48 (0.27)	93.63 (0.32)	84.43 (1.38)	72.18 (0.43)	36.33 (0.70)	76.84 (0.58)	61.37 (1.83)
Qwen								
Echo	63.80 (0.66)	40.28 (1.62)	85.19 (0.35)	65.80 (0.97)	64.57 (0.35)	28.49 (0.68)	62.19 (0.64)	42.57 (1.63)
SPILL (Qwen)	73.66 (0.66)	53.44 (1.57)	90.62 (0.22)	75.73 (0.99)	68.15 (0.27)	31.45 (0.74)	69.06 (0.34)	49.09 (1.68)
Summarizer	64.80 (0.35)	42.92 (1.41)	91.55 (0.18)	77.54 (0.94)	76.33 (0.48)	39.08 (0.90)	76.43 (0.89)	61.91 (2.35)
SPILL (Qwen)	70.98 (0.22)	49.94 (0.67)	94.10 (0.19)	85.02 (1.04)	77.41 (0.24)	42.45 (1.42)	78.12(0.40)	65.33 (1.27)
Llama								
Echo	68.40 (0.46)	46.20 (0.46)	87.03 (0.47)	70.60 (0.79)	68.19 (0.48)	31.49 (1.24)	61.62 (0.76)	42.24 (1.45)
SPILL (Llama)	73.44 (0.35)	53.50 (0.60)	90.49 (0.29)	78.89 (0.47)	70.53 (0.27)	33.55 (1.19)	67.42 (0.64)	47.57 (1.82)
Summarizer	67.47 (0.21)	43.99 (1.19)	92.49 (0.31)	81.26 (1.27)	76.51 (0.19)	40.10 (0.86)	74.67 (0.66)	59.23 (1.62)
SPILL (Llama)	70.31 (0.20)	48.62 (0.99)	93.59 (0.17)	86.12 (0.96)	76.26 (0.48)	40.59 (1.25)	76.20 (0.47)	63.19 (1.77)
Gemma								
Echo	71.20 (0.45)	50.32 (1.88)	90.13 (0.24)	73.36 (0.65)	71.24 (0.18)	32.82 (0.80)	70.51 (0.93)	50.13 (0.89)
SPILL (Gemma)	79.37 (0.51)	60.25 (1.94)	93.77 (0.09)	82.97 (1.32)	74.74 (0.34)	38.70 (1.38)	76.24 (0.40)	58.75 (0.73)
Summarizer	69.74 (0.28)	47.16 (1.14)	94.10 (0.24)	83.67 (0.74)	78.90 (0.32)	**45.14** (1.87)	77.83 (0.58)	63.48 (2.22)
SPILL (Gemma)	75.38 (0.22)	55.12 (0.65)	**95.49** (0.08)	**88.25** (0.70)	**79.01** (0.48)	43.77 (1.04)	**79.11** (0.57)	64.33 (2.54)

Table 4: Results for the four benchmark sets. Scores are averages over 5 runs, with standard deviations shown in parentheses. Model names in bold denote the embedding model, with names in parentheses indicating the LLM used. Plain refers to directly using the embedding for clustering. Boldface numbers highlight the highest values globally, while underlined values indicate the best within each embedder. Results for ClusterLLM, KeyphraseClust, and SynCID are taken directly from their papers.

hypothesize that reduced performance occurs because some pooled utterances come from different clusters than the seed utterance.

To verify this, Table 6 shows a hypothetical result with 100% correct selection rate. In this scenario, we assume that the correct candidates in the first stage are already known, without relying on an LLM for validation. Under this assumption, the results show consistent improvements over directly using the embedder, which supports our hypothesis, and aligns with the simulation result.

6.3 Qualitative analysis

We apply t-SNE (Van der Maaten and Hinton, 2008) to reduce the embedding dimensions for 2D visualization. The embeddings are obtained from **Gemma** with **Echo** prompt. We compare the results between the **Echo** and SPILL. From Figure 1 is clear that our method separates the clusters better. However, we also observe that for the **Mtop** dataset, both approaches struggle with clustering it well. As an additional analysis, we randomly select examples from the first and second stage selections using Gemma-Echo on the Bank77 dataset. In the first stage, we observe that utterances with different intents tend to appear as they become farther from the seed. Second, the LLM can identify utterances with the same intent as the seed. Finally, we find that the LLM is able to select more distant same-cluster utterances from the seed, thereby introducing greater variety. The examples are provided in Appendix G.

7 Conclusions

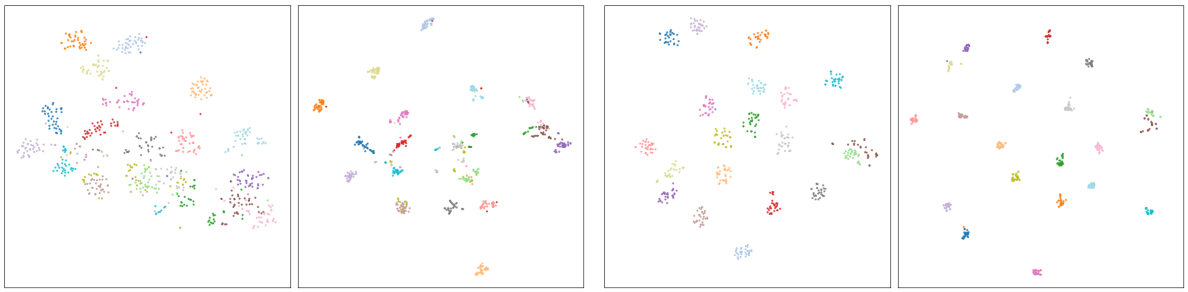
We proposed Selection and Pooling with Large Language Models (SPILL), an intuitive, zero-shot method for intent clustering without fine-tuning. SPILL is applicable to any embedder and does not require fine-tuning. We found that (1) our proposed method enables viewing a clustering task as a small-scale selection problem, providing a novel perspective on the clustering process; (2) our

	Bank 77		Clinc150		Mtop		Massive	
	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc
E5-large								
Plain	77.19 (0.34)	59.60 (1.42)	91.27 (0.38)	75.92 (0.91)	70.87 (0.23)	34.21 (0.57)	71.38 (0.55)	53.85 (1.28)
w\1st stage	79.79 (0.24)	66.16 (0.80)	91.49 (0.31)	81.59 (0.69)	70.62 (0.53)	33.37 (0.93)	71.32 (0.26)	55.70 (1.13)
w\2nd stage (Gemma)	83.56 (0.47)	70.25 (1.56)	92.93 (0.08)	83.18 (0.78)	71.77 (0.35)	36.83 (1.10)	75.40 (0.51)	60.28 (1.81)
w\2nd stage (Llama)	80.94 (0.29)	66.84 (0.97)	91.41 (0.07)	80.09 (1.31)	70.52 (0.51)	35.04 (1.06)	72.74 (0.44)	58.06 (1.52)
w\2nd stage (Qwen)	83.64 (0.26)	70.31 (0.75)	92.67 (0.31)	82.58 (0.87)	71.20 (0.36)	36.48 (0.73)	74.46 (0.57)	59.38 (1.20)
Gemma								
Echo	71.20 (0.45)	50.32 (1.88)	90.13 (0.24)	73.36 (0.65)	71.24 (0.18)	32.82 (0.80)	70.51 (0.93)	50.13 (0.89)
w\1st stage	72.09 (0.39)	51.55 (0.94)	90.75 (0.17)	79.46 (0.41)	71.70 (0.42)	34.17 (1.13)	70.45 (0.33)	49.17 (0.54)
w\2nd stage	79.37 (0.51)	60.25 (1.94)	93.77 (0.09)	82.97 (1.32)	74.74 (0.34)	38.70 (1.38)	76.24 (0.40)	58.75 (0.73)
Summarizer	69.74 (0.28)	47.16 (1.14)	94.10 (0.24)	83.67 (0.74)	78.90 (0.32)	45.14 (1.87)	77.83 (0.58)	63.48 (2.22)
w\1st stage	70.98 (0.36)	49.84 (0.65)	94.28 (0.08)	86.64 (0.41)	77.89 (0.55)	41.65 (1.31)	77.82 (0.25)	63.38 (1.14)
w\2nd stage	75.38 (0.22)	55.12 (0.65)	95.49 (0.08)	88.25 (0.70)	79.01 (0.48)	43.77 (1.04)	79.11 (0.57)	64.33 (2.54)

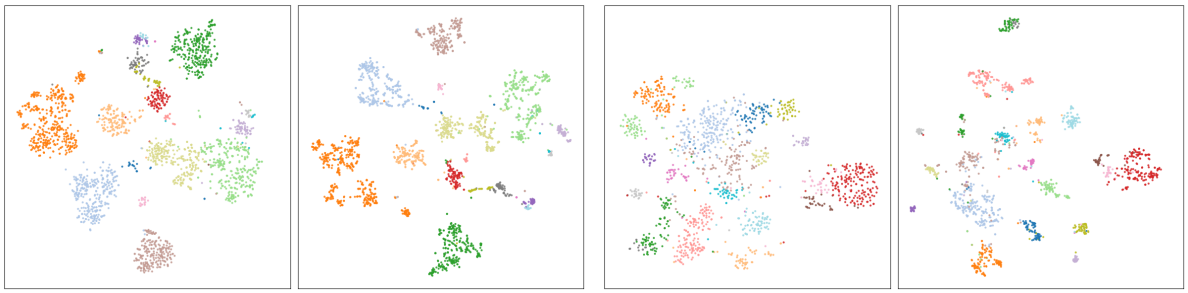
Table 5: Ablation study: Contribution of the first and selection stage (%). Average over 5 runs. The results of other embedders is in appendix F.

	Bank 77		Clinc150		Mtop		Massive	
	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc
E5-large								
Plain	77.19 (0.34)	59.60 (1.42)	91.27 (0.38)	75.92 (0.91)	70.87 (0.23)	34.21 (0.57)	71.38 (0.55)	53.85 (1.28)
Ground truth	91.74 (0.15)	83.40 (0.97)	98.17 (0.17)	94.28 (1.11)	81.87 (0.50)	45.44 (1.65)	87.08 (0.39)	73.16 (1.87)
Gemma								
Echo	71.20 (0.45)	50.32 (1.88)	90.13 (0.24)	73.36 (0.65)	71.24 (0.18)	32.82 (0.80)	70.51 (0.93)	50.13 (0.89)
Ground truth	86.47 (0.46)	71.49 (0.88)	97.76 (0.19)	92.21 (0.85)	82.06 (0.62)	45.74 (1.85)	86.37 (0.54)	69.00 (1.76)
Sum	69.74 (0.28)	47.16 (1.14)	94.10 (0.24)	83.67 (0.74)	78.90 (0.32)	45.14 (1.87)	77.83 (0.58)	63.48 (2.22)
Ground truth	81.05 (0.33)	62.23 (1.59)	97.68 (0.09)	92.67 (0.46)	83.65 (0.44)	49.84 (1.17)	85.94 (0.40)	73.34 (1.14)

Table 6: Average results of ground truth pooling (%) over 5 runs. Ground truth pools each seed utterance with same-cluster utterances from 20 candidates in the stage one. Results for other embedders are in Appendix F.



(a) Bank77 (b) Clinc150



(c) Mtop (d) Massive

Figure 1: T-SNE plots for the four datasets (20 clusters for each). Left: Echo (Gemma). Right: SPILL

method demonstrates general improvements regardless of the embedding model used, highlighting its versatility; (3) Our method can achieve comparable results to other SOTA research studies without fine-tuning and by using smaller models; (4) It proves to be effective in data-limited domains (zero-shot setting), showing its adaptability in challenging scenarios. For future work, we see potential for improved prompting, the addition of few-shot settings, and generalisation to other languages.

Limitations

Language. Like most of the prior work, we only focus on English utterance datasets. This is relevant because most of benchmark datasets are in English.

Prompt design Our research uses only one prompt in the second stage across different models and datasets. Our main goal is to avoid overdoing prompt engineering and to make our approach generalizable. The prompt simply explains the task and ensures the desired answer format for easy extraction. However, this also implies there still be room for improvement in our results because we did not specifically design a unique prompt for each LLM or dataset.

Few-shot settings. Our research only considers the zero-shot setting. However, in our second stage selection, we use LLMs for selection. If LLMs are provided with some examples with known intents, they could potentially identify the relevance between seed and candidate utterances more effectively. This exploration will be left for future work.

References

AI@Meta. 2024. Llama 3 model card.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Maarten De Raedt, Fréderic Godin, Thomas Demeester, and Chris Develder. 2023. Idas: Intent discovery with abstractive summarization. In *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pages 71–88.

Bingzhu Du, Nan Su, Yuchi Zhang, and Yongliang Wang. 2023. A two-stage progressive intent clustering for task-oriented dialogue. In *Proceedings of The Eleventh Dialog System Technology Challenge*, pages 48–56.

James Gung, Raphael Shu, Emily Moeng, Wesley Rose, Salvatore Romeo, Yassine Benajiba, Arshit Gupta, Saab Mansour, and Yi Zhang. 2023. Intent induction from conversations for task-oriented dialogue track at dstc 11. *arXiv preprint arXiv:2304.12982*.

Peihao Huang, Yan Huang, Wei Wang, and Liang Wang. 2014. Deep embedding network for clustering. In *2014 22nd International conference on pattern recognition*, pages 1532–1537. IEEE.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2023. Scaling sentence embeddings with large language models. *arXiv preprint arXiv:2307.16645*.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316.

Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. 2020. Contrastive representation learning: A framework and review. *Ieee Access*, 8:193907–193934.

Yibin Lei, Di Wu, Tianyi Zhou, Tao Shen, Yu Cao, Chongyang Tao, and Andrew Yates. 2024. Meta-task prompting elicits embedding from large language models. *arXiv preprint arXiv:2402.18458*.

Jinggui Liang and Lizi Liao. 2023. Clusterprompt: Cluster semantic enhanced prompt learning for new intent discovery. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10468–10481.

Jinggui Liang, Lizi Liao, Hao Fei, and Jing Jiang. 2024. Synergizing large language models and pre-trained smaller models for conversational intent discovery. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14133–14147.

I-Fan Lin, Faegheh Hasibi, and Suzan Verberne. 2024. Generate then refine: Data augmentation for zero-shot intent detection. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13138–13146.

S. Lloyd. 1982. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Marina Meilă. 2007. Comparing clusterings—an information based distance. *Journal of multivariate analysis*, 98(5):873–895.

Yutao Mou, Keqing He, Pei Wang, Yanan Wu, Jingang Wang, Wei Wu, and Weiran Xu. 2022a. Watch the neighbors: A unified k-nearest neighbor contrastive learning framework for ood intent discovery. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1517–1529.

Yutao Mou, Keqing He, Yanan Wu, Zhiyuan Zeng, Hong Xu, Huixing Jiang, Wei Wu, and Weiran Xu. 2022b. Disentangled knowledge transfer for ood intent discovery with unified contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 46–53.

Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, and Erik Cambria. 2023. Recent advances in deep learning based dialogue systems: A systematic survey. *Artificial intelligence review*, 56(4):3055–3155.

William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Juan A Rodriguez, Nicholas Botzer, David Vazquez, Christopher Pal, Marco Pedersoli, and Issam Laradji. 2024. Intentgpt: Few-shot intent discovery with large language models. *arXiv preprint arXiv:2411.10670*.

AB Siddique, Fuad Jamour, Luxun Xu, and Vagelis Hristidis. 2021. Generalized zero-shot intent detection via commonsense knowledge. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1925–1929.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*.

Gemma Team. 2024a. Gemma.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Vijay Viswanathan, Kiril Gashteovski, Kiril Gashteovski, Carolin Lawrence, Tongshuang Wu, and Graham Neubig. 2024. Large language models enable few-shot clustering. *Transactions of the Association for Computational Linguistics*, 12:321–333.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen Mckeown, Ramesh Nallapati, Andrew O Arnold, and Bing Xiang. 2021a. Supporting clustering with contrastive learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5419–5430.

Hanlei Zhang, Hua Xu, Ting-En Lin, and Rui Lyu. 2021b. Discovering new intents with deep aligned clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14365–14373.

Hanlei Zhang, Hua Xu, Xin Wang, Fei Long, and Kai Gao. 2023a. A clustering framework for unsupervised and semi-supervised new intent discovery. *IEEE Transactions on Knowledge and Data Engineering*.

Yuwei Zhang, Zihan Wang, and Jingbo Shang. 2023b. Clusterllm: Large language models as a guide for text clustering. *arXiv preprint arXiv:2305.14871*.

Yuwei Zhang, Haode Zhang, Li-Ming Zhan, Xiao-Ming Wu, and Albert Lam. 2022. New intent discovery with pre-training and contrastive learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 256–269.

Yunhua Zhou, Guofeng Quan, and Xipeng Qiu. 2023. A probabilistic framework for discovering new intents. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3771–3784.

A Mathematical proof of the inequality

Proof. To show the inequality holds, we first to show both $\overset{*}{Z}_{ih}$ and Z_{ih} have the same mean:

$$E(\overset{*}{Z}_{ih}) = E\left(\frac{Z_{ih} + \sum_{m=1}^{k} Z_{imh}}{1+k}\right) \tag{3}$$

$$= \frac{1}{(1+k)}\left\{E(Z_{ih}) + \sum_{m=1}^{k} E(Z_{imh})\right\} \tag{4}$$

$$= \frac{1}{(1+k)}(\mu_h + k\mu_h) \tag{5}$$

$$= \mu_h \tag{6}$$

Z_{imh} is the hth dimension element of $\mathbf{Z}_{im}$ (see Eq. 1). Since $E[\overset{*}{Z}_{ih}] = E[Z_{ih}] = \mu_h$, the inequality (see the formula 2) can be rewritten as follows:

$$\sum_{h=1}^{d} \text{Var}[\overset{*}{Z}_{ih}] - \text{Var}[Z_{ih}] < 0 \tag{7}$$

Then, we have:

$$\text{Var}[\overset{*}{Z}_{ih}] = \text{Var}\left[\frac{Z_{ih} + \sum_{m=1}^{k} Z_{imh}}{1+k}\right]$$

$$\overset{\diamond}{=} \frac{1}{(1+k)^2}\left[\text{Var}[Z_{ih}] + \sum_{m=1}^{k} \text{Var}[Z_{imh}]\right]$$

$$= \frac{1}{(1+k)^2}[\sigma_h^2 + k \cdot \sigma_h^2]$$

$$= \frac{1}{1+k} \cdot \sigma_h^2 < \sigma_h^2 = \text{Var}[Z_{ih}]$$

□

Note that the step marked with $\diamond$ holds due to the independence of these random variables. Since this property holds for each dimension, the entire summation (see Eq. 7) will hold as well.

B Simulation process

We consider 3 clusters in a 128-dimensional space. The mean and variances are averaged over 128 dimensions. For each experiment trial, between 50 and 250 data points are independently sampled for each cluster. The data is drawn from either a normal distribution or a skewed log-normal distribution. For the normal distribution, the μ and σ^2 in each dimension are sampled from the ranges $(0, 1e-10)$ and $(20, 60)$, respectively. The log-normal distribution is generated by taking the exponential of data points sampled from the normal distribution, with the σ^2 and μ in each dimension sampled from the ranges $(0, 1e-10)$ and $(1.5, 2)$.

Summarizer (Jiang et al., 2023):
The task is intent detection. The goal is to identify the purpose or goal behind a user input. The user intent of this sentence: {sentences} means in one word:"
Echo (Repetition) (Springer et al., 2024):
Instruct: The task is intent detection. The goal is to identify the purpose or goal behind a user input. Give the user intent of the utterance,
User utterances:{sentences} User utterances again:{sentences}
Instructor (Su et al., 2022) and (Liang and Liao, 2023):
Bank77: Represent the bank purpose for retrieval:
Mtop: Represent the sentence for retrieval:
Clinc150 and Massive: Represent the sentence for retrieving the purpose:

Table 7: Task Instructions Prompt for the first stage. Note that for **Summarizer** and **Echo**, we use the same prompt across different datasets.

C Simulation with varied dimensionality

Figure 2 shows **TopK** pooling performs even better in higher dimension than **Rd** when the cluster is skewed. For a normal distribution, **Rd** consistently outperforms **TopK**.

D Prompt used in the first stage

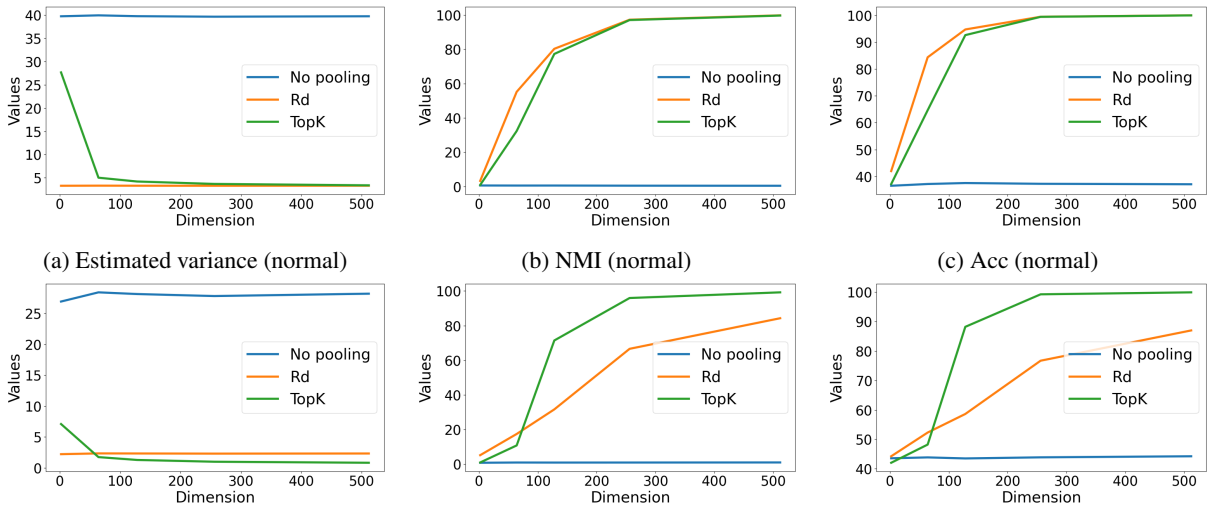
Table 7 shows the prompts we apply. Specifically, we follow the main prompt structures outlined in **Echo** and **Summarizer**, with minor modifications to include wording tailored to our specific task of intent detection. For **Instructor**, we adopt the same prompt used in the study we reference for comparison (Zhang et al., 2023b), which is similar to the ones used in the original study (Su et al., 2022). For **E5-large**, we follow their practice to add 'Query:' as prefix.

E Hyperparameter selection process

We use **Instructor** as our first-stage embedding model to align with previous research, such as Zhang et al. (2023b) and Viswanathan et al. (2024), who proposed methods built on this embedder. For the second stage, while both studies use **GPT**, we instead use the smaller and open-source model Llama for the reproducibility and accessibility.

Although a higher tolerance gives us a better chance to cover more utterances from the same cluster, this also increases the computation required for the second stage. We set $L = 20$.

We experiment with different combinations of $l_{top} \in \{2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$ utterances.



(a) Estimated variance (normal) (b) NMI (normal) (c) Acc (normal)

(d) Estimated Variance (log-normal) (e) NMI (log-normal) (f) Acc (log-normal)

Figure 2: Average variance, clustering accuracy (Acc), and NMI over 50 runs with different dimensions given $k = 10$. We only present variance from one of the three clusters (they all have the same trend) for readability. The simulation process is the same as table 1, and only varies in dimension.

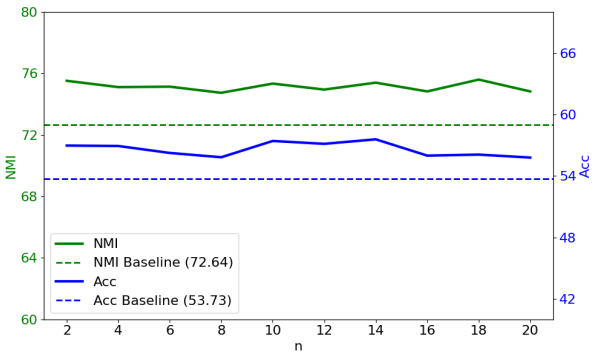


Figure 3: The average clustering accuracy (Acc) and NMI for different values of l_{top} over 5 runs. The dashed line refers to the result obtained by directly using the embedder.

Figure 3 shows that the $(l_{top}, l_{random} = (14, 6)$ overall gives the best performance.

F Ablation analysis for all encoder and decoder embedders

Here, we provide the complete results of the ablation study. Details of the performance contributions in both the first and selection stages can be found in table 8. The average correct ratio and the number of selected items are presented in table 9, while the details of the ground truth analysis can be found in table 10.

G Some examples of Selection

We randomly select three examples extracted from the first selection using **Gemma-Echo** and our method **Gemma-SPILL** from the **Bank77** dataset. In the first stage, the selection is ordered by the distance from the seed (starting at 15, derived through chunk sampling). First, we observe that in the initial stage, utterances with different intents appear in the later part of the selection. Second, it can be seen that the LLM effectively selects utterances with the same intent as the seed. Additionally, the LLM is capable of correctly selecting utterances from the later part of the list, introducing more variety to the pool while maintaining consistency with the seed. For instance, in Example 1, the furthest utterance is selected by the LLM without any mistakes in selecting incorrect utterances.

Example 1:
Seed utterance: I would appreciate it if I could get an item refunded (intent: request refund)

Gemma-Echo-1stStage
1. Would I be able to get a refund for something I bought? (request refund)
2. I don't want the item, I bought it on accident, can I get a refund? (request refund)
3. I would like a refund for something I bought (request refund)

	Bank 77		Clinc150		Mtop		Massive	
	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc
E5-large								
Plain	77.19 (0.34)	59.60 (1.42)	91.27 (0.38)	75.92 (0.91)	70.87 (0.23)	34.21 (0.57)	71.38 (0.55)	53.85 (1.28)
w\1st stage	79.79 (0.24)	66.16 (0.80)	91.49 (0.31)	81.59 (0.69)	70.62 (0.53)	33.37 (0.93)	71.32 (0.26)	55.70 (1.13)
w\2nd stage (Gemma)	83.56 (0.47)	70.25 (1.56)	92.93 (0.08)	83.18 (0.78)	71.77 (0.35)	36.83 (1.10)	75.40 (0.51)	60.28 (1.81)
w\2nd stage (Llama)	80.94 (0.29)	66.84 (0.97)	91.41 (0.07)	80.09 (1.31)	70.52 (0.51)	35.04 (1.06)	72.74 (0.44)	58.06 (1.52)
w\2nd stage (Qwen)	83.64 (0.26)	70.31 (0.75)	92.67 (0.31)	82.58 (0.87)	71.20 (0.36)	36.48 (0.73)	74.46 (0.57)	59.38 (1.20)
Instructor-large								
Plain	82.38 (0.59)	65.70 (1.78)	93.25 (0.32)	81.12 (2.27)	71.69 (0.60)	34.06 (2.51)	74.56 (0.37)	56.62 (1.75)
w\1st stage	82.83 (0.45)	68.69 (1.80)	93.48 (0.17)	86.18 (0.72)	71.65 (0.34)	32.63 (1.27)	75.29 (0.59)	59.35 (3.06)
w\2nd stage (Gemma)	85.01 (0.29)	71.05 (0.83)	93.77 (0.35)	85.14 (1.04)	72.65 (0.32)	37.11 (1.62)	77.62 (0.46)	62.42 (2.06)
w\2nd stage (Llama)	83.37 (0.22)	69.55 (0.60)	92.96 (0.18)	84.31 (0.72)	71.41 (0.31)	35.18 (1.12)	75.28 (0.83)	58.79 (2.75)
w\2nd stage (Qwen)	85.12 (0.30)	71.48 (0.27)	93.63 (0.32)	84.43 (1.38)	72.18 (0.43)	36.33 (0.70)	76.84 (0.58)	61.37 (1.83)
Qewn								
Echo	63.80 (0.66)	40.28 (1.62)	85.19 (0.35)	65.80 (0.97)	64.57 (0.35)	28.49 (0.68)	62.19 (0.64)	42.57 (1.63)
w\1st stage	65.10 (0.50)	42.84 (0.85)	87.06 (0.25)	70.95 (1.05)	65.79 (0.13)	29.43 (0.26)	63.88 (0.58)	43.69 (1.73)
w\2nd stage	73.66 (0.66)	53.44 (1.57)	90.62 (0.22)	75.73 (0.99)	68.15 (0.27)	31.45 (0.74)	69.06 (0.34)	49.09 (1.68)
Summarizer	64.80 (0.35)	42.92 (1.41)	91.55 (0.18)	77.54 (0.94)	76.33 (0.48)	39.08 (0.90)	76.43 (0.89)	61.91 (2.35)
w\1st stage	65.21 (0.39)	43.95 (0.47)	92.35 (0.15)	83.69 (0.56)	76.13 (0.25)	39.21 (1.11)	76.66 (0.49)	63.67 (2.03)
w\2nd stage	70.98 (0.22)	49.94 (0.67)	94.10 (0.19)	85.02 (1.04)	77.41 (0.24)	42.45 (1.42)	78.12 (0.40)	65.33 (1.27)
Llama								
Echo	68.40 (0.46)	46.20 (0.46)	87.03 (0.47)	70.60 (0.79)	68.19 (0.48)	31.49 (1.24)	61.62 (0.76)	42.24 (1.45)
w\1st stage	70.25 (0.56)	48.88 (1.03)	87.57 (0.14)	74.09 (0.48)	68.91 (0.42)	32.07 (0.49)	63.19 (0.40)	42.97 (0.92)
w\2nd stage	73.44 (0.35)	53.50 (0.60)	90.49 (0.29)	78.89 (0.47)	70.53 (0.27)	33.55 (1.19)	67.42 (0.64)	47.57 (1.82)
Summarizer	67.47 (0.21)	43.99 (1.19)	92.49 (0.31)	81.26 (1.27)	76.51 (0.19)	40.10 (0.86)	74.67 (0.66)	59.23 (1.62)
w\1st stage	68.54 (0.34)	46.11 (0.77)	93.15 (0.11)	85.55 (0.23)	75.87 (0.32)	39.41 (1.34)	75.81 (0.33)	63.30 (1.82)
w\2nd stage	70.31 (0.20)	48.62 (0.99)	93.59 (0.17)	86.12 (0.96)	76.26 (0.48)	40.59 (1.25)	76.20 (0.47)	63.19 (1.77)
Gemma								
Echo	71.20 (0.45)	50.32 (1.88)	90.13 (0.24)	73.36 (0.65)	71.24 (0.18)	32.82 (0.80)	70.51 (0.93)	50.13 (0.89)
w\1st stage	72.09 (0.39)	51.55 (0.94)	90.75 (0.17)	79.46 (0.41)	71.70 (0.42)	34.17 (1.13)	70.45 (0.33)	49.17 (0.54)
w\2nd stage	79.37 (0.51)	60.25 (1.94)	93.77 (0.09)	82.97 (1.32)	74.74 (0.34)	38.70 (1.38)	76.24 (0.40)	58.75 (0.73)
Summarizer	69.74 (0.28)	47.16 (1.14)	94.10 (0.24)	83.67 (0.74)	78.90 (0.32)	45.14 (1.87)	77.83 (0.58)	63.48 (2.22)
w\1st stage	70.98 (0.36)	49.84 (0.65)	94.28 (0.08)	86.64 (0.41)	77.89 (0.55)	41.65 (1.31)	77.82 (0.25)	63.38 (1.14)
w\2nd stage	75.38 (0.22)	55.12 (0.65)	95.49 (0.08)	88.25 (0.70)	79.01 (0.48)	43.77 (1.04)	79.11 (0.57)	64.33 (2.54)

Table 8: Contribution of the first and selection stage (%). Average over 5 runs.

4. I want to get an item refunded (request refund)
5. I bought something but now I would like a refund. How do I do that? (request refund)
6. I am not Happy with this product can i get a refund? (request refund)
7. Can I have an item refunded? (request refund)
8. I bought this item and was charged the wrong amount can I get a refund? (request refund)
9. I am unhappy with my purchase, how do I cancel the order? (request refund)
10. I would like to cancel a payment. I purchased something several days ago and i still have not received it. (request refund)
11. I would like to know why I was charged twice for my purchase. (transaction charged twice)
12. I would like to know why my payment is still pending, can you help? (pending transfer)
13. Hello! I recently made a purchase and I'm needing to cancel my order and process a refund as soon as possible. (request refund)
14. I would like a refund on one of your products that has been sold to me (request refund)
15. Can I receive a refund for my item? (request refund)
16. Can I have a refund? (request refund)
17. I want to reverse a purchase. Can I cancel it? (request refund)
18. I need a refund on an item I have not received. Am I able to simply cancel the payment? I don't know how to do this. (request refund)
19. I requested a refund, and never received it. What can I do? (Refund not showing up)
20. Hi there! I need to cancel an order I recently made and start processing a refund. Can you please help me with this and set up the refund as soon as possible? It's very urgent. (request refund)

Gemma-Echo-2ndStage

1. Can I have a refund? (request refund)
2. Can I receive a refund for my item? (request refund)
3. Would I be able to get a refund for something I bought? (request refund)
4. I would like a refund on one of your products that has been sold to me (request refund)
5. I am not Happy with this product can i get a refund? (request refund)
6. I need a refund on an item I have not received. Am I able to simply cancel the payment? I don't know how to do this. (request refund)
7. I want to get an item refunded (request refund)
8. I would like a refund for something I bought (request refund)
9. Hi there! I need to cancel an order I recently

	Bank 77		Clinc150		Mtop		Massive	
	Ratio(%)	# Selection	Ratio(%)	# Selection	Ratio(%)	# Selection	Ratio(%)	# Selection
E5-large								
Plain w\1st stage	62.01	20.00	71.90	20.00	71.56	20.00	59.12	20.00
w\2nd stage (Gemma)	80.30	10.54	93.05	8.05	84.26	5.91	80.88	6.12
w\2nd stage (Llama)	71.14	12.13	86.52	9.50	80.12	7.09	73.42	7.38
w\2nd stage (Qwen)	80.52	8.61	90.80	7.14	82.45	4.72	78.65	5.17
Instructor-large								
w\1st stage	67.84	20.00	78.17	20.00	75.41	20.00	65.38	20.00
w\2nd stage (Gemma)	80.66	11.28	93.43	8.15	85.46	5.94	81.85	6.27
w\2nd stage (Llama)	73.48	13.12	88.79	9.89	81.98	7.26	76.34	7.73
w\2nd stage (Qwen)	81.12	9.41	92.06	7.31	84.50	4.82	79.70	5.38
Qwen								
Echo								
w\1st stage	44.11	20.00	65.69	20.00	70.18	20.00	56.66	20.00
w\2st stage	72.99	6.94	90.22	6.24	83.29	3.84	78.96	4.25
Summarizer								
w\1st stage	43.44	20.00	78.49	20.00	83.28	20.00	71.47	20.00
w\2st stage	69.95	7.30	93.03	6.72	89.15	4.17	82.78	4.94
Llama								
Echo								
w\1st stage	48.89	20.00	66.84	20.00	73.51	20.00	55.20	20.00
w\2nd stage	61.90	10.54	85.84	8.41	82.83	6.23	71.70	6.32
Summarizer								
w\1st stage	47.85	20.00	81.76	20.00	83.46	20.00	71.66	20.00
w\2nd stage	57.70	11.54	89.90	9.58	87.09	6.97	79.37	7.42
Gemma								
Echo								
w\1st stage	50.85	20.00	71.15	20.00	77.01	20.00	61.25	20.00
w\2nd stage	75.12	9.60	93.92	7.54	88.46	4.94	82.02	5.45
Summarizer								
w\1st stage	52.11	20.00	84.68	20.00	85.34	20.00	74.49	20.00
w\2nd stage	73.00	10.10	95.32	8.01	90.19	5.43	84.60	6.13

Table 9: Average correct selection ratio and average number of utterances selected over 5 runs. Note: the number for the first stage remains constant as $L = 20$ is fixed. The ratio is defined as the number of same-cluster utterances out of the total number of selections.

made and start processing a refund. Can you please help me with this and set up the refund as soon as possible? It's very urgent. (request refund)
10. I bought this item and was charged the wrong amount can I get a refund? (request refund)
11. I don't want the item, I bought it on accident, can I get a refund? (request refund)
12. Can I have an item refunded? (request refund)
13. I bought something but now I would like a refund. How do I do that? (request refund)

Example 2:
Seed utterance: When getting my ID checked, what are the steps involved? (intent: verify my identity)

Gemma-Echo-1stStage
1. What kind of documents do I need for the identity check? (verify my identity)
2. What will I need for identity verification? (verify my identity)
3. Are there any documents needed for the identity check? (verify my identity)
4. What's the process for ID verification? (verify my identity)
5. Is there any documentation needed for the identity check? (verify my identity)
6. What are the steps I need to take to verify my identity? (verify my identity)
7. Do I need any kind of documentation for the identity check? (verify my identity)
8. What do I do for the identity check? (verify my identity)
9. What is needed to prove my identity? (verify my identity)
10. Let me know the steps for the identity checks (verify my identity)
11. What is the need to verify my identity? (why verify identity)
12. What are the steps to verify my identity? (verify my identity)
13. What all am I required to show for the identity check? (verify my identity)
14. What things do I need to verify my identity? (verify my identity)
15. Is there a specific type you need for identity verification? (verify my identity)
16. What do I need to do to verify the source of my funds? (verify source of funds)
17. What's with not verifying my Id? (unable to verify identity)
18. I need the source of my funds verified. How do I do this? (verify source of funds)
19. What do i need to verify my id? (unable to verify identity)
20. What other methods are there to verify my

	Bank 77		Clinc150		Mtop		Massive	
	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc
E5-large								
Plain	77.19 (0.34)	59.60 (1.42)	91.27 (0.38)	75.92 (0.91)	70.87 (0.23)	34.21 (0.57)	71.38 (0.55)	53.85 (1.28)
Ground truth	91.74 (0.15)	83.40 (0.97)	98.17 (0.17)	94.28 (1.11)	81.87 (0.50)	45.44 (1.65)	87.08 (0.39)	73.16 (1.87)
Instructor-large								
Plain	82.38 (0.59)	65.70 (1.78)	93.25 (0.32)	81.12 (2.27)	71.69 (0.60)	34.06 (2.51)	74.56 (0.37)	56.62 (1.75)
Ground truth	92.68 (0.32)	84.54 (0.91)	98.54 (0.10)	95.44 (0.40)	81.29 (0.27)	43.69 (1.10)	87.30 (0.35)	72.04 (1.87)
Qwen								
Echo	63.80 (0.66)	40.28 (1.62)	85.19 (0.35)	65.80 (0.97)	64.57 (0.35)	28.49 (0.68)	62.19 (0.64)	42.57 (1.63)
Ground truth	81.64 (0.68)	64.21 (1.87)	95.60 (0.15)	87.15 (0.68)	76.14 (0.42)	38.35 (1.25)	79.49 (0.46)	60.56 (1.22)
Sum	64.80 (0.35)	42.92 (1.41)	91.55 (0.18)	77.54 (0.94)	76.33 (0.48)	39.08 (0.90)	76.43 (0.89)	61.91 (2.35)
Ground truth	77.05 (0.56)	57.58 (0.79)	97.33 (0.17)	91.91 (0.71)	83.19 (0.40)	49.35 (1.15)	86.99 (0.33)	74.72 (0.90)
Llama								
Echo	68.40 (0.46)	46.20 (0.46)	87.03 (0.47)	70.60 (0.79)	68.19 (0.48)	31.49 (1.24)	61.62 (0.76)	42.24 (1.45)
Ground truth	85.44 (0.39)	68.84 (0.82)	95.74 (0.11)	88.45 (0.75)	79.27 (0.28)	40.69 (0.79)	78.74 (0.79)	59.48 (2.75)
Sum	67.47 (0.21)	43.99 (1.19)	92.49 (0.31)	81.26 (1.27)	76.51 (0.19)	40.10 (0.86)	74.67 (0.66)	59.23 (1.62)
Ground truth	80.08 (0.30)	59.68 (0.74)	97.56 (0.08)	93.41 (0.62)	82.77 (0.29)	48.53 (0.79)	85.66 (0.64)	72.53 (2.05)
Gemma								
Echo	71.20 (0.45)	50.32 (1.88)	90.13 (0.24)	73.36 (0.65)	71.24 (0.18)	32.82 (0.80)	70.51 (0.93)	50.13 (0.89)
Ground truth	86.47 (0.46)	71.49 (0.88)	97.76 (0.19)	92.21 (0.85)	82.06 (0.62)	45.74 (1.85)	86.37 (0.54)	69.00 (1.76)
Sum	69.74 (0.28)	47.16 (1.14)	94.10 (0.24)	83.67 (0.74)	78.90 (0.32)	45.14 (1.87)	77.83 (0.58)	63.48 (2.22)
Ground truth	81.05 (0.33)	62.23 (1.59)	97.68 (0.09)	92.67 (0.46)	83.65 (0.44)	49.84 (1.17)	85.94 (0.40)	73.34 (1.14)

Table 10: Average results of ground truth pooling (%) over 5 runs. Ground truth refers to each seed utterance being pooled with other same-cluster (i.e. always selected correctly from a pool of 20 candidates) utterances from the first-stage collection.

identity? (why verify identity)

Gemma-Echo-2ndStage

1. What are the steps I need to take to verify my identity? (verify my identity)
2. What's the process for ID verification? (verify my identity)
3. Let me know the steps for the identity checks (verify my identity)
4. What are the steps to verify my identity? (verify my identity)