The Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI-25)

CDW-CoT: Clustered Distance-Weighted Chain-of-Thoughts Reasoning

Yuanheng Fang[1], Guoqing Chao*[1], Wenqiang Lei[2], Shaobo Li[1], Dianhui Chu[1]

[1]Harbin Institute of Technology, Weihai, 264209, Shandong, China
[2]Sichuan University, Chengdu, 610065, Sichuan, China
23s130443@stu.hit.edu.cn, guoqingchao@hit.edu.cn, wenqianglei@scu.edu.cn, lishaobo@hit.edu.cn, chudh@hit.edu.cn

Abstract

Large Language Models (LLMs) have recently achieved impressive results in complex reasoning tasks through Chain of Thought (CoT) prompting. However, most existing CoT methods rely on using the same prompts, whether manually designed or automatically generated, to handle the entire dataset. This one-size-fits-all approach may fail to meet the specific needs arising from the diversities within a single dataset. To solve this problem, we propose the Clustered Distance-Weighted Chain of Thought (CDW-CoT) method, which dynamically constructs prompts tailored to the characteristics of each data instance by integrating clustering and prompt optimization techniques. Our method employs clustering algorithms to categorize the dataset into distinct groups, from which a candidate pool of prompts is selected to reflect the inherent diversity within the dataset. For each cluster, CDW-CoT trains the optimal prompt probability distribution tailored to their specific characteristics. Finally, it dynamically constructs a unique prompt probability distribution for each test instance, based on its proximity to cluster centers, from which prompts are selected for reasoning. CDW-CoT consistently outperforms traditional CoT methods across six datasets, including commonsense, symbolic, and mathematical reasoning tasks. Specifically, when compared to manual CoT, CDW-CoT achieves an average accuracy improvement of 25.34% on LLaMA2 (13B) and 15.72% on LLaMA3 (8B).

Introduction

Recent advancements in LLMs, such as GPT-3 (Brown et al. 2020), LLama2 (Touvron et al. 2023), and Llama3 (Dubey et al. 2024), have significantly enhanced their capability to tackle complex reasoning tasks. Some studies (Brown et al. 2020; Thoppilan et al. 2022) have demonstrated LLMs' impressive performance in decomposing multi-step problems into manageable intermediate steps, resulting in more accurate and contextually relevant answers. A technique that has gained prominence in this context is CoT prompting, which systematically structures the reasoning process into a series of intermediate steps. This method has been shown to significantly improve the model's performance on complex tasks across various domains (Wei et al. 2022).

Initially, CoT prompting involved embedding manually crafted exemplars within the model's prompt to guide its reasoning process—a method that was effective but labor-intensive and not scalable (Wei et al. 2022). This approach evolved into Zero-Shot-CoT (Kojima et al. 2022), which allowed models to engage in reasoning without task-specific exemplars, relying on generic prompts to elicit intermediate reasoning steps. However, the absence of tailored guidance often limited its efficacy in more complex or domain-specific tasks.

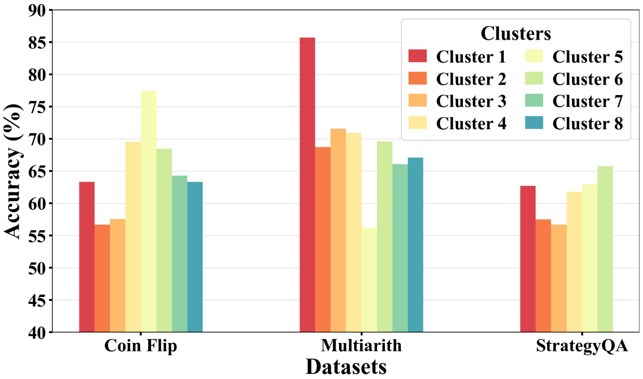

Figure 1: Using the same prompts for all instances in the dataset resulted in significant performance variability across different clusters, highlighting the limitations of Auto-CoT in addressing diverse reasoning demands within different data categories. This underscores the need for tailored prompt strategies.

To address the intensive manual efforts required by Manual-CoT, the Auto-CoT paradigm has been proposed (Zhang et al. 2022). This method automates the generation of reasoning chains by clustering related questions and selecting a representative question from each cluster to generate a reasoning chain using simple heuristics. Recent works (Chu et al. 2023) focus on further automating (Shum, Diao, and Zhang 2023) and refining the CoT generation process. Techniques such as Enhanced Reasoning (Wang et al. 2023; Li and Qiu 2023; Wu, Zhang, and Huang 2023), Voting and Ranking (Fu et al. 2022; Li et al. 2023, 2022), and Verification and Refinement (Lyu et al. 2023; Shao

*Corresponding author

et al. 2023; Aggarwal et al. 2023; Weng et al. 2022; Wang et al. 2022a) have been developed to enhance the quality and applicability of CoT across diverse tasks. Especially, Automate-CoT approach integrates variance-reduced policy gradient methods to optimize the selection of CoT exemplars, significantly reducing the dependency on manual prompt engineering.

Despite these advancements, Automatic Chain of Thought Prompting still encounters significant challenges, particularly because most of them use the same prompts for all instances in the dataset. As exemplified by Auto-CoT (Zhang et al. 2022) shown in Figure 1, this approach results in considerable performance variability across different clusters. This variability highlights its inability to effectively address the diverse reasoning demands of different data categories, emphasizing the necessity for more adaptive techniques that can tailor prompts to the unique characteristics of each cluster.

To address the limitations inherent in manual and uniform prompt strategies across diverse datasets, we introduce the CDW-CoT framework. This method innovatively combines clustering with dynamic prompt optimization to enhance the adaptability and precision in reasoning tasks. By segmenting the dataset into distinct clusters, CDW-CoT harnesses the unique characteristics of each group to generate a tailored prompt candidate pool. For each cluster, we calculate an optimal prompt probability distribution, finely tuned to the specific demands and nuances of the data. Additionally, our framework incorporates a distance-weighted prompt selection mechanism that dynamically adapts reasoning strategies based on the proximity of test instances to the cluster centers. This ensures that each reasoning step is contextually informed and effectively customized, significantly improving the reasoning accuracy. Experiment results on six datasets show the superiority of our proposed method CDW-CoT over the state-of-the-art methods.

The main contributions of our work are summarized as follows:

- We leverage the clustering technique to produce a diverse prompt candidate pool that mining the category-specific information sufficiently, enhancing the relevance and effectiveness of prompts for different clusters within the same dataset.
- Our framework calculates the optimal prompt probability distributions for each cluster within the dataset, effectively treating datasets as distinct clusters and enabling highly targeted reasoning approaches tailored to the unique characteristics of each group.
- We introduce a method for employing distance-weighted calculations for each test instance's prompt probability distribution, which refines and tailors the reasoning process of large language models to the specific requirements of each instance.
- Our empirical evaluations confirm that the CDW-CoT framework substantially outperforms traditional CoT methods, achieving the state-of-the-art accuracy across multiple datasets.

Related Works

Chain of Thought Prompting

CoT Prompting enhances logical reasoning in LLMs like GPT-3, developed as the models increased in scale (Brown et al. 2020). Wei et al. first introduced CoT, using manually constructed detailed prompts to systematically guide LLMs through each logical step, significantly enhancing reasoning transparency and accuracy (Wei et al. 2022). Building on this foundational work, Zero-Shot-CoT employs the simple prompt "Let's think step by step" to facilitate unsupervised reasoning, effectively enabling CoT without predefined examples (Kojima et al. 2022).

Automatic Chain of Thought Prompting

Addressing the accuracy challenges in Zero-Shot-CoT and the resource intensity of Few-Shot CoT, Auto-CoT automates reasoning chain generation. This method clusters related questions, using cluster centers as prompts, thereby reducing manual labor and improving scalability (Zhang et al. 2022). Building on this, complexity-based prompting selects prompts based on their reasoning complexity, which has been shown to improve performance on multi-step reasoning tasks significantly (Fu et al. 2022). Furthermore, self-verification techniques introduced in studies allow models to cross-check and refine their outputs (Weng et al. 2022). In the context of mathematical reasoning, the MathPrompter framework validates results by leveraging different algebraic expressions or Python functions to solve problems (Imani, Du, and Shrivastava 2023).

Policy Gradient Optimization Methods

The Black-Box Discrete Prompt Learning (BDPL) employs variance-reduced policy gradients to optimize prompts efficiently, enhancing LLM performance without direct access to model parameters (Diao et al. 2022). Following this, the Black-Box Prompt Optimization (BPO) further refines these advancements by aligning LLM outputs with user preferences through optimized prompts, improving user interactions and satisfaction (Cheng et al. 2023). Dynamic Prompt Learning via Policy Gradient (PROMPTPG) further refines this approach by dynamically selecting in-context examples that optimize reasoning tasks, particularly in complex domains like mathematics (Lu et al. 2022). Building on these strategies, the Automatic Prompt Augmentation and Selection method extends the application of policy gradient methods to CoT prompting, automating both the generation and the optimization of reasoning chains (Shum, Diao, and Zhang 2023).

CDW-CoT Model

In this section, we introduce our proposed CDW-CoT model, as depicted in Figure 2. The CDW-CoT is composed of the three components: cluster-based prompt candidate pool initialization, optimizing prompt probability distributions for clusters and distance-weighted prompt selection and inference.

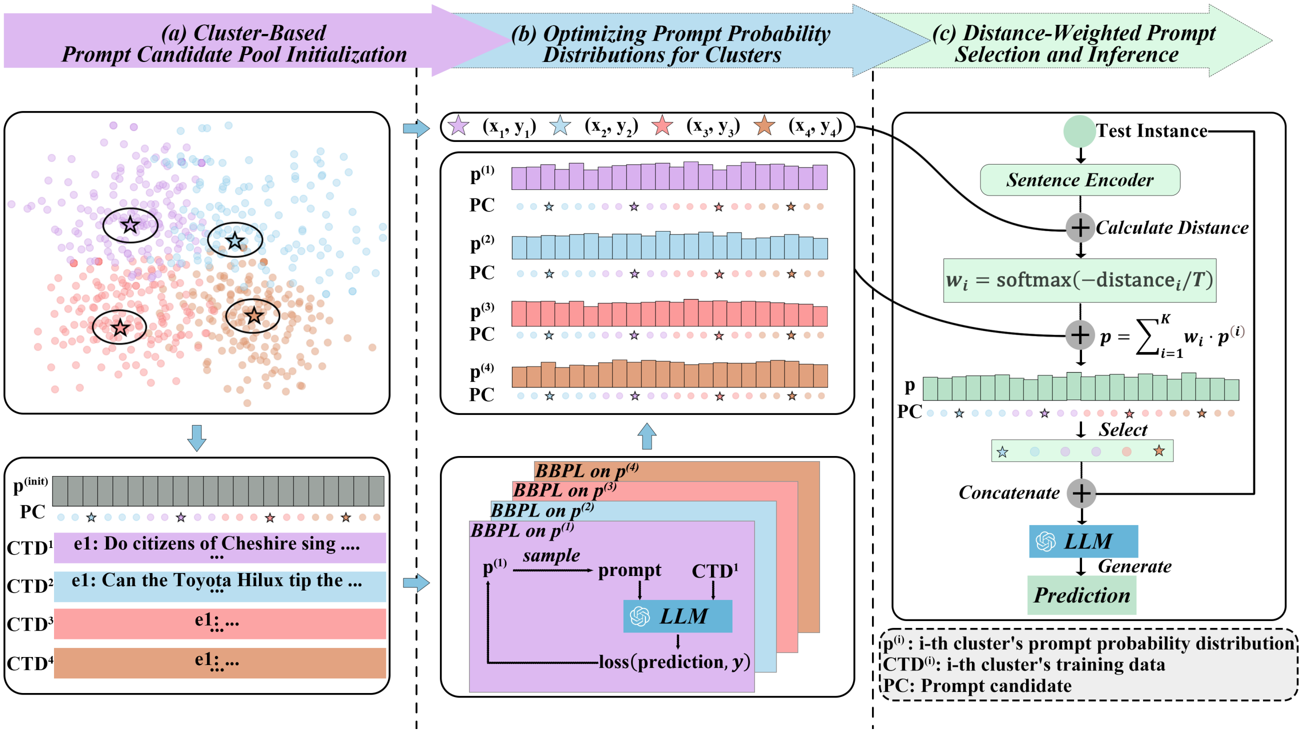
(a) Cluster-Based Prompt Candidate Pool Initialization
(b) Optimizing Prompt Probability Distributions for Clusters
(c) Distance-Weighted Prompt Selection and Inference
$p^{(init)}$
PC
CTD1 e1: Do citizens of Cheshire sing ....
CTD2 e1: Can the Toyota Hilux tip the ...
CTD3 e1: ...
CTD4 e1: ...
(x_1, y_1) (x_2, y_2) (x_3, y_3) (x_4, y_4)
$p^{(1)}$ PC
$p^{(2)}$ PC
$p^{(3)}$ PC
$p^{(4)}$ PC
BBPL on $p^{(4)}$
BBPL on $p^{(3)}$
BBPL on $p^{(2)}$
BBPL on $p^{(1)}$
$p^{(1)}$ sample prompt CTD1
LLM
loss(prediction, y)
Test Instance
Sentence Encoder
Calculate Distance
$w_i = \mathrm{softmax}(-\mathrm{distance}_i/T)$
$p = \sum_{i=1}^{K} w_i \cdot p^{(i)}$
p
PC
Select
Concatenate
LLM
Generate
Prediction
$p^{(i)}$: i-th cluster's prompt probability distribution
$CTD^{(i)}$: i-th cluster's training data
PC: Prompt candidate

Figure 2: Framework of the proposed CDW-CoT. (a) After clustering, prompt candidates are generated based on the cluster centers. CTD$^{(i)}$ and cluster center coordinates (x_i, y_i) are also obtained. (b) For each cluster, $p^{(i)}$ is initially set to $p^{(\text{init})}$ and then optimized through Black-Box Prompt Learning (BBPL) to achieve the optimal distribution. (c) For test instance, a distance-weighted prompt probability distribution is constructed to select prompts and perform reasoning.

Cluster-Based Prompt Candidate Pool Initialization

The dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ consists of N question-answer pairs. Each instance x_i is transformed into the vector embeddings $\{e_i\}$ using a pre-trained sentence transformer and clustered into K groups via K-means.

As illustrated in Figure 2(a), following the clustering process, the preliminary selection of prompt candidates begins from the centroid of each cluster. The number of candidates selected from each cluster S_c is based on the proportion of that cluster's data within the overall dataset. Once the preliminary prompt candidate pool is established, it is refined into the final prompt candidates (PC) through zero-shot-CoT (Kojima et al. 2022). The entire process is detailed in Algorithm 1.

Simultaneously, we establish an initial prompt probability distribution $p^{(\text{init})}$, where each candidate in the pool is assigned an equal probability. This balanced distribution, along with the cluster-specific training data CTD$^{(i)}$, serves as the foundation for the next phase of the model training process.

Furthermore, the coordinates of each cluster's centroid, obtained during the clustering process, are stored to use for calculating the distance of the test instance to them. These coordinates play a critical role in the model's final phase: distance-weighted prompt selection and inference, where they guide the distance-weighted prompt probability distribution and ensure the model adapts effectively to new, unseen instances.

Algorithm 1: Cluster-Based Prompt Candidate Pool Initialization

Input: $X = \{x_1, \ldots, x_N\}$, $Y = \{y_1, \ldots, y_N\}$, Number of Clusters K, Pool Size S
Output: Prompt Candidates PC

```
$\{e_i\} \leftarrow \mathrm{SentenceTransformer}(X)$
cluster_assignment $\leftarrow$ K-Means$(K, \{e_i\})$
Initialize cluster data structure $C = [\,]$
for $i \leftarrow 1$ to $N$ do
    $c_i \leftarrow$ cluster_assignment$[i]$
    $d \leftarrow$ EuclideanDistance$(e_i,$ cluster_centers$[c_i])$
    Add $(x_i, y_i, d)$ to $C[c_i]$
end for
Prepare to build prompt candidate pool:
for $c \leftarrow 1$ to $K$ do
    Compute $S_c \leftarrow \left\lfloor \frac{|C[c]|}{N} \times S \right\rfloor$
    Sort $C[c]$ by distance $d$
    $P_c \leftarrow$ SelectTop$(C[c], \text{size} = S_c)$
    Add $P_c$ to $P$
end for
$PC \leftarrow$ ZeroShotCoT$(P, \text{LLM})$
return $PC$
```

Optimizing Prompt Probability Distributions for Clusters

As illustrated in Figure 2(b), the optimization of prompt probability distribution for each cluster is conducted using the BBPL method. The process begins by setting the initial distribution $p^{(i)}$ for each cluster to a uniform distribution $p^{(\text{init})}$. This distribution is then refined through gradient descent, based on feedback from the training process with $\text{CTD}^{(i)}$.

For each cluster i, prompts are sampled according to the $p^{(i)}$. These prompts, along with the $\text{CTD}^{(i)}$, are input into the LLM, which returns the prediction and computes the corresponding loss.

The gradient for each prompt is computed as:

$$\delta = -\frac{1}{p^{(i)}}, \tag{1}$$

where $p^{(i)}$ represents the prompt probability matrix for cluster i.

These gradients are adjusted based on the actual usage of prompts during training:

$$\delta_{k,m,n} = \begin{cases} -\delta_{k,m,n} & \text{if } n \text{ is selected} \\ \delta_{k,m,n} & \text{otherwise.} \end{cases} \tag{2}$$

Here, k indexes the sample within the batch, m represents the prompt, and n corresponds to the indices of the prompts sampled. Adjustments are weighted by the deviation of each sample's loss from the batch average:

$$\text{Gradient} = \sum_{k=1}^{\text{s}} \frac{L_k - L_{\text{avg}}}{\text{s} - 1} \times \delta_k, \tag{3}$$

where s is the sample size used in the optimization process. Using the aggregated gradient and learning rate η, the probability matrix $p^{(i)}$ is updated according to the following formula:

$$p^{(i)}_{mn} \leftarrow p^{(i)}_{mn} - \eta \cdot \text{Gradient}_{mn}. \tag{4}$$

Probabilities are then normalized and clipped within the range [0,1] to ensure stability:

$$p^{(i)}_{mn} \leftarrow \max(\min(p^{(i)}_{mn}, 1), 0). \tag{5}$$

The optimized prompt probabilities are validated on a validation dataset. If the performance improves, these updated settings are used for future operations.

Through this process, we obtain the optimal prompt probability distribution for each cluster, as depicted in Figure 2(b).

Distance-Weighted Prompt Selection and Inference

This subsection describes how we construct unique prompt probability distribution for each test instance through distance weighting, using the optimal prompt probability distribution for each cluster, and the coordinates of cluster centers. The obtained prompt probability distribution is used to select the final prompts that are concatenated with test instance and input into the LLM, as illustrated in Figure 2(c).

Distance Calculation For each test instance, its embedding obtained through a sentence Transformer is compared with the cluster centers to compute the Euclidean distances. These distances reflect the instance's similarity to each cluster.

Weight Calculation The computed distances are converted into weights using a temperature-scaled softmax function:

$$\text{weights} = \frac{\exp(-\text{distances}/T)}{\sum \exp(-\text{distances}/T)}, \tag{6}$$

where T is a temperature parameter that controls the sensitivity to distance variations.

Prompt Distribution Calculation The prompt probability distribution for the test instance is calculated by weighting the optimal prompt distributions of each cluster:

$$\text{p} = \sum_{i=1}^{K} \text{weights}_i \cdot p^{(i)}, \tag{7}$$

where K is the number of clusters. This weighted combination tailors the prompt probability distribution to the specific characteristics of the test instance.

Query Execution and Evaluation Prompts are selected based on the computed distribution and then concatenated with the original test instance. The LLM uses this concatenated input to generate a response, which is subsequently evaluated against the actual answer to assess the accuracy and effectiveness.

The steps of this process are outlined in Algorithm 2, demonstrating the implementation of distance-weighted prompt probability distribution and its impact on inference.

Algorithm 2: Distance-Weighted Prompt Selection and Inference

Input: Test dataset D_{test}, Cluster centers C, Temperature T, Optimized prompt probabilities for each cluster $p^{(i)}$
Output: Evaluated responses R

```
Initialize weights as an empty list
Initialize $R$ as an empty list to store responses
for each instance $q$ in $D_{\text{test}}$ do
    $e_q \leftarrow$ SentenceTransformer($q$)
    for each center $c$ in $C$ do
        Compute distance: distance $=$ Euclidean$(e_q, c)$
        Compute weight: weight $= \exp(-\text{distance}/T)$
        Append weight to weights
    end for
    Normalize weights: weights $\leftarrow \frac{\text{weights}}{\sum \text{weights}}$
    Compute the prompt probability distribution for q: p $= \sum_{i=1}^{K} \text{weights}_i \cdot p^{(i)}$
    Select prompt using p
    Input prompt and $q$ into LLM to generate response
    Evaluate response accuracy and append to $R$
end for
return $R$
```

Method	Commonsense Reasoning		Symbolic Reasoning		Mathematical Reasoning	
	CSQA	StrategyQA	Letter	Coin	MultiArith	AQuA
			LLaMA2 (13B)			
Zero-Shot-CoT (Kojima et al. 2022)	32.68	48.41	30.20	51.80	71.00	30.31
Auto-CoT (Zhang et al. 2022)	51.09	56.24	30.80	51.00	44.17	24.02
Manual-CoT (Wei et al. 2022)	46.52	60.48	15.80	47.60	44.17	30.31
CDW-CoT (ours)	**61.41**	**70.06**	**82.67**	**61.33**	**85.56**	**35.89**
			LLaMA3 (8B)			
Zero-Shot-CoT (Kojima et al. 2022)	60.52	66.72	76.67	44.40	90.00	48.03
Auto-CoT (Zhang et al. 2022)	69.57	60.57	50.60	60.80	68.83	31.10
Manual-CoT (Wei et al. 2022)	56.84	57.51	61.40	60.20	85.17	32.68
CDW-CoT (ours)	**72.15**	**67.44**	**82.67**	**70.70**	**95.17**	**58.97**

Table 1: Comparative exact match accuracy across various datasets using LLaMA2 (13B) and LLaMA3 (8B) models. The CDW-CoT method consistently outperforms traditional CoT methods in all tested reasoning tasks and datasets, improving accuracy for both models.

Experiments and Results

Experiments Setup

Tasks and Datasets We evaluated the CoT frameworks on six datasets across three categories of reasoning tasks, which are listed as follows:

- **Commonsense Reasoning**:

  CommonsenseQA (CSQA) (Talmor et al. 2018): A widely used dataset for evaluating commonsense reasoning through multiple-choice questions that require inferencing based on prior knowledge and context.

  StrategyQA (Geva et al. 2021): It contains questions requiring implicit multi-hop reasoning to derive yes/no answers, testing the model's ability to connect various pieces of information logically.

- **Symbolic Reasoning**:

  Letter (Wei et al. 2022): It involves tasks like last letter concatenation, designed to test the symbolic reasoning capabilities of models.

  Coin (Wei et al. 2022): It focuses on determining the state of a coin after a series of flips, evaluating the model's ability to track state changes through symbolic manipulations.

- **Mathematical Reasoning**:

  MultiArith (Roy and Roth 2016): It consists of multi-step arithmetic word problems that require a sequence of operations to reach the solution, testing multi-step reasoning in arithmetic contexts.

  AQuA (Ling et al. 2017): It includes complex arithmetic word problems with multiple-choice answers, providing a benchmark for evaluating sophisticated reasoning and calculation skills.

Models and Baselines We conducted comparative experiments using both the LLaMA2 (13B) and LLaMA3 (8B) models, running on two NVIDIA 4090 GPUs locally. The LLaMA2 (13B) model was selected for its easy-use, while LLaMA3 (8B) was chosen to evaluate the scalability of our approach across different large language models. To evaluate the performance of our CDW-CoT framework, we compared it against several baseline methods implemented on the same LLM:

- **Zero-Shot-CoT** (Kojima et al. 2022): It uses a simple prompt like "Let's think step by step" without requiring prior demonstrations.
- **Auto-CoT** (Zhang et al. 2022): It automates reasoning chain generation by clustering similar questions and and using the cluster centers as prompts.
- **Manual-CoT** (Wei et al. 2022): It involves crafting manually designed reasoning chains, tailored with specific demonstrations for each dataset.

To show the superiority of our method, the number of prompts used in our work was kept less than or equal to those used in other methods, since more prompts typically yield better performance. The num of prompts used in the CDW-CoT framework were: 6 (CommonsenseQA), 5 (StrategyQA), 4 (Letter), 6 (Coin), 5 (MultiArith), and 4 (AQuA).

Data Split and Number of Clusters Identification Datasets were divided into training, evaluation, and test subsets with proportions of approximately 60%, 25%, and 15%, respectively (Wang et al. 2022b). After dividing the data, we identified the number of clusters according to the Auto-CoT setup, and then adjusted the number of clusters for certain datasets from the default 8 to 3, as shown in Table 2.

Dataset	**Total**	**Train**	**Eval**	**Test**	**#Clusters**
CSQA	1,221	725	312	184	7
StrategyQA	2,290	1,362	584	344	6
Letter	500	297	128	75	4
Coin	500	297	128	75	3
MultiArith	600	357	153	90	3
AQuA	254	150	65	39	4

Table 2: Data Split and Number of Clusters Statistics.

Prompt Engineering Configuring prompts effectively is crucial for training models across diverse datasets. This phase involved three key parameters:

- **Pool Size**: We maintained a consistent pool of 40 potential prompts for each dataset to enable thorough exploration of diverse reasoning pathways.
- **Sample Size**: During training, each instance was tested against five unique prompt combinations, assessing the effectiveness of various configurations.
- **Temperature**: A temperature of 0.3 was used to optimize prompt selection during testing.

Our primary metric, exact match accuracy, measures the degree responses correctly answer the instances across various reasoning domains. As detailed in Table 1, our results demonstrate substantial performance improvements across all the tasks and both models used, underscoring the effectiveness of the CDW-CoT framework. For both the LLaMA2 (13B) and LLaMA3 (8B) models, we compared our method against the best baseline method among the three we evaluated.

Main Results

CDW-CoT consistently outperforms traditional CoT methods across various reasoning tasks and datasets, improving accuracy for both LLaMA2 and LLaMA3 models. The detailed results are as follows:

Commonsense Reasoning: For CommonsenseQA, CDW-CoT improved exact match accuracy by 10.32% (51.09% → 61.41%) on LLaMA2 (13B) and by 2.58% (69.57% → 72.15%) on LLaMA3 (8B). For StrategyQA, CDW-CoT increased accuracy by 9.58% (60.48% → 70.06%) on LLaMA2 (13B) and by 0.72% (66.72% → 67.44%) on LLaMA3 (8B).

Symbolic Reasoning: In the Letter dataset, CDW-CoT significantly improved accuracy by 51.87% (30.80% → 82.67%) on LLaMA2 (13B) and by 6.07% (76.67% → 82.67%) on LLaMA3 (8B). In the Coin dataset, CDW-CoT improved accuracy by 9.53% (51.80% → 61.33%) on LLaMA2 (13B) and by 9.90% (60.80% → 70.70%) on LLaMA3 (8B).

Mathematical Reasoning: CDW-CoT recorded a 14.56% increase (71.00% → 85.56%) on MultiArith with LLaMA2 (13B) and a 5.17% increase (90.00% → 95.17%) on LLaMA3 (8B). For AQuA, accuracy improved by 5.58% (30.31% → 35.89%) on LLaMA2 (13B) and by 10.94% (48.03% → 58.97%) on LLaMA3 (8B).

These results demonstrate that the CDW-CoT framework effectively enhances performance across a wide range of reasoning tasks, including commonsense reasoning, mathematical reasoning, and symbolic reasoning. The framework consistently outperforms Zero-Shot-CoT, Manual-CoT and even Auto-CoT and shows significant improvements across different LLMs, as confirmed by the results in Table 1.

Ablation Study

To evaluate the effectiveness of each component of our model, we conduct the experiments with different model versions by removing the corresponding component.

The three model versions are described as follows:

- **Distance Weighting (Dist-W)**: This version implements the complete model, using clustering to generate optimal prompt probability distributions tailored to each category. It adjusts the reasoning process for each test instance by employing distance-weighted prompt distribution, enhancing specificity based on proximity to cluster centers.
- **Nearest Cluster (Near-C)**: This streamlined approach assigns the nearest cluster's prompt distribution to each test instance, omitting the computational complexity of distance weighting. This method emphasizes efficiency while still utilizing the benefits of clustering.
- **No Clustering (No-Clust)**: This baseline approach without clustering phase uses a single, global optimal prompt probability distribution, derived from the entire dataset and applied uniformly across all test instances.

The effectiveness of each model version was assessed using the same setups with the main experiments.

Dataset	Dist-W	Near-C	No-Clust
CSQA	**61.41**	53.26	60.33
StrategyQA	**70.06**	67.44	67.15
Letter	**82.67**	81.33	81.11
Coin	**61.33**	58.67	56.00
MultiArith	**85.56**	77.78	77.78
AQuA	**35.89**	28.21	23.08

Table 3: Ablation study of different model versions across datasets, showing percentage accuracies.

The results of our ablation study, as shown in Table 3, clearly demonstrate the effectiveness of each component in the CDW-CoT framework. The important role of the Dist-W method is evident, as it consistently achieves the highest accuracy across all the datasets. This method highlights the importance of clustering and distance-based prompt optimization, allowing the model to adapt its reasoning pathways effectively by considering the unique aspects of each test instance. The Distance Weighting method is particularly successful in complex tasks such as MultiArith and Letter, where precise and context-aware reasoning is crucial.

The Near-C model, which only relies on the nearest cluster's prompt distribution without distance weighting, is limited in its capability to effectively use the optimal prompt probability distributions across multiple clusters. This constraint leads to a 5.04% decrease across datasets averagely, as shown in Table 3.

The No-Clust model uses a uniform prompt distribution for all the instances, which reduces its effectiveness. Its lower performance in Table 3, with an average decrease of approximately 5.24% compared to the full model, highlights the importance of constructing category-specific prompt distributions to address the distinct demands of various data categories effectively.

This ablation study confirms the robustness of our CDW-CoT framework, demonstrating that each component, particularly clustering and distance weighting, plays a crucial role in enhancing the reasoning performance.

Sensitivity Analysis of Temperature

Our analysis investigates the impact of the temperature parameter T in our framework.

We explored the effects of temperature settings ranging from 0.1 to 1.0 measured with accuracy. The experiments were conducted using the LLaMA2(13B) model on StrategyQA and MultiArith.

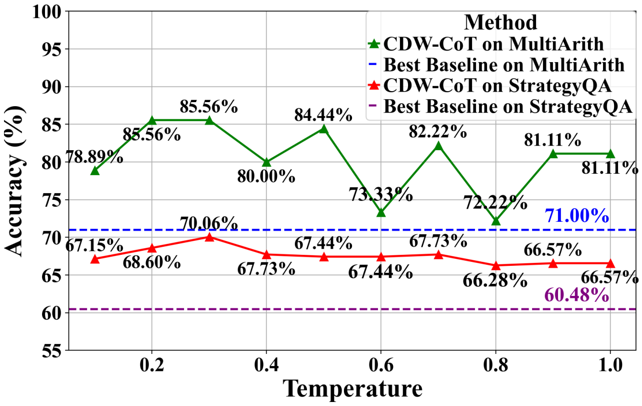

Figure 3: Sensitivity analysis of temperature for CDW-CoT on StrategyQA and MultiArith datasets.

Temperature plays a pivotal role in the CDW-CoT framework, as evidenced by our detailed results depicted in Figure 3. At a lower temperature of 0.1, the model becomes overly sensitive, disproportionately focusing on the nearest cluster even when it may not be the most relevant. This excessive sensitivity often leads to inaccuracies, especially when the query is ambiguously positioned relative to multiple clusters.

Conversely, at a temperature setting of 1.0, the model's performance declines due to an overly generalized approach that incorporates too much irrelevant cluster information. This almost uniform focus reduces the response accuracy and fails to fully leverage the optimal prompt distributions for each cluster.

Throughout all the temperatures, the CDW-CoT consistently surpasses the best baseline among the conventional methods compared, highlighting its superior reasoning capabilities and robust adaptability. The model achieves optimal performance at a temperature of 0.3, striking an effective balance between specificity and sensitivity. This setting allows the model to accurately concentrate on the most pertinent cluster features, thus maximizing the accuracy and maintaining the flexibility across a variety of reasoning tasks.

Impact of Pool Size on CDW-CoT

Similar to the sensitivity analysis of temperature effects, we make the analysis to explore the impact of Pool Size S on the CDW-CoT framework. This parameter is important as it controls the number of prompt candidates extracted from clusters.

We varied the pool size from 10 to 40 to assess how the quantity of available prompt candidates impacts the model's performance. This investigation was conducted using the LLaMA2(13B) model on two datasets: CommonsenseQA and MultiArith.

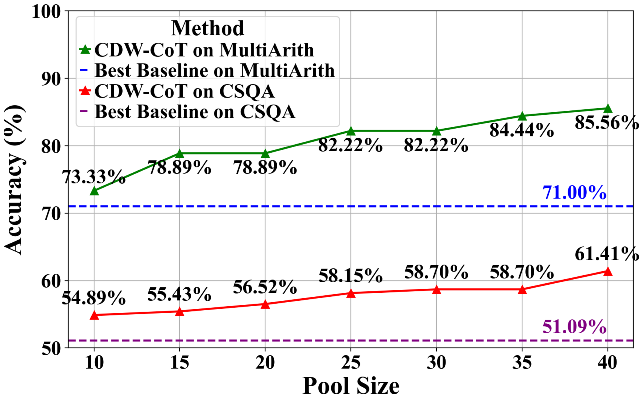

Figure 4: Impact of pool size on CDW-CoT on CommonsenseQA and MultiArith datasets.

Based on the analysis and trends from Figure 4, we observe that increasing the pool size consistently enhances the model's performance across both datasets. As expected, a larger candidate pool allows the CDW-CoT framework to better explore diverse reasoning paths. For MultiArith, accuracy steadily improves from 73.33% at a pool size of 10 to 85.56% at a pool size of 40. Similarly, for CommonsenseQA, accuracy increases from 54.89% to 61.41% as the pool size grows.

While a larger pool increases reasoning diversity and improves the accuracy, it also increases the computational costs. In our experiments, we chose a pool size of 40 as an optimal balance between performance gains and efficiency. This selection ensures that the CDW-CoT framework achieves high accuracy across different reasoning tasks without incurring excessive computational overhead, effectively balancing decision quality and resource use.

Conclusion

In this paper, we propose a novel CoT method named CDW-CoT to enhance the adaptability and accuracy of LLMs in complex reasoning tasks. Our method introduces the clustering to categorize the datasets into tailored prompt pools, improving the representative ability to diverse data characteristics. It calculates an optimal prompt probability distribution for each cluster, enabling targeted reasoning that aligns with its unique characteristics. By designing the distance-weighted prompt selection, CDW-CoT dynamically adjusts the reasoning strategies based on the proximity to cluster centers, demonstrating superior performance over traditional methods across six datasets. Future work includes reducing computational overhead and extending applicability to multimodal tasks like image-text reasoning.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (No. 62276079), Young Teacher Development Fund of Harbin Institute of Technology IDGA10002071, Research and Innovation Foundation of Harbin Institute of Technology IDGAZMZ00210325 and the Special Funding Program of Shandong Taishan Scholars Project.

References

Aggarwal, P.; Madaan, A.; Yang, Y.; et al. 2023. Let's Sample Step by Step: Adaptive-Consistency for Efficient Reasoning and Coding with LLMs. *arXiv preprint arXiv:2305.11860*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cheng, J.; Liu, X.; Zheng, K.; Ke, P.; Wang, H.; Dong, Y.; Tang, J.; and Huang, M. 2023. Black-box prompt optimization: Aligning large language models without model training. *arXiv preprint arXiv:2311.04155*.

Chu, Z.; Chen, J.; Chen, Q.; Yu, W.; He, T.; Wang, H.; Peng, W.; Liu, M.; Qin, B.; and Liu, T. 2023. A survey of chain of thought reasoning: Advances, frontiers and future. *arXiv preprint arXiv:2309.15402*.

Diao, S.; Huang, Z.; Xu, R.; Li, X.; Lin, Y.; Zhou, X.; and Zhang, T. 2022. Black-box prompt learning for pre-trained language models. *arXiv preprint arXiv:2201.08531*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Fu, Y.; Peng, H.; Sabharwal, A.; Clark, P.; and Khot, T. 2022. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations*.

Geva, M.; Khashabi, D.; Segal, E.; Khot, T.; Roth, D.; and Berant, J. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9: 346–361.

Imani, S.; Du, L.; and Shrivastava, H. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Kojima, T.; Gu, S. S.; Reid, M.; Matsuo, Y.; and Iwasawa, Y. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213.

Li, X.; and Qiu, X. 2023. Mot: Memory-of-thought enables chatgpt to self-improve. *arXiv preprint arXiv:2305.05181*.

Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.-G.; and Chen, W. 2022. On the advance of making language models better reasoners. *arXiv preprint arXiv:2206.02336*.

Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.-G.; and Chen, W. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5315–5333.

Ling, W.; Yogatama, D.; Dyer, C.; and Blunsom, P. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Lu, P.; Qiu, L.; Chang, K.-W.; Wu, Y. N.; Zhu, S.-C.; Rajpurohit, T.; Clark, P.; and Kalyan, A. 2022. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610*.

Lyu, Q.; Havaldar, S.; Stein, A.; Zhang, L.; Rao, D.; Wong, E.; Apidianaki, M.; and Callison-Burch, C. 2023. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.

Roy, S.; and Roth, D. 2016. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*.

Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Synthetic prompting: Generating chain-of-thought demonstrations for large language models. In *International Conference on Machine Learning*, 30706–30775. PMLR.

Shum, K.; Diao, S.; and Zhang, T. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. *arXiv preprint arXiv:2302.12822*.

Talmor, A.; Herzig, J.; Lourie, N.; and Berant, J. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Thoppilan, R.; De Freitas, D.; Hall, J.; Shazeer, N.; Kulshreshtha, A.; Cheng, H.-T.; Jin, A.; Bos, T.; Baker, L.; Du, Y.; et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, J.; Sun, Q.; Li, X.; and Gao, M. 2023. Boosting language models reasoning with chain-of-knowledge prompting. *arXiv preprint arXiv:2306.06427*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2022a. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; and Zhou, D. 2022b. Rationale-augmented ensembles in language models. *arXiv preprint arXiv:2207.00747*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Weng, Y.; Zhu, M.; Xia, F.; Li, B.; He, S.; Liu, S.; Sun, B.; Liu, K.; and Zhao, J. 2022. Large language models are better reasoners with self-verification. *arXiv preprint arXiv:2212.09561*.

Wu, D.; Zhang, J.; and Huang, X. 2023. Chain of thought prompting elicits knowledge augmentation. *arXiv preprint arXiv:2307.01640*.

Zhang, Z.; Zhang, A.; Li, M.; and Smola, A. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.