Collaborative Stance Detection via Small-Large Language Model Consistency Verification

Yu Yan[1,3], Sheng Sun[1], Zixiang Tang[2], Teli Liu[2], and Min Liu[1,3,4](✉)*

[1] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{yanyu24z,sunsheng,liumin}@ict.ac.cn
[2] People Public Security University of China, Beijing, China
{202220250034,202221610039}@stu.ppsuc.edu.cn
[3] University of Chinese Academy of Sciences, Beijing, China
[4] Zhongguancun Laboratory, Beijing, China

Abstract. Stance detection on social media aims to identify attitudes expressed in tweets towards specific targets. Current studies prioritize Large Language Models (LLMs) over Small Language Models (SLMs) due to the overwhelming performance improving provided by LLMs. However, heavily relying on LLMs for stance detection, regardless of the cost, is impractical for real-world social media monitoring systems that require vast data analysis. To this end, we propose **Co**llaborative Stance Detection via Small-Large Language Model Consistency **Ver**ification (**CoVer**) framework, which enhances LLM utilization via context-shared batch reasoning and logical verification between LLM and SLM. Specifically, instead of processing each text individually, CoVer processes texts batch-by-batch, obtaining stance predictions and corresponding explanations via LLM reasoning in a shared context. Then, to exclude the bias caused by context noises, CoVer introduces the SLM for logical consistency verification. Finally, texts that repeatedly exhibit low logical consistency are classified using consistency-weighted aggregation of prior LLM stance predictions. Our experiments show that CoVer outperforms state-of-the-art methods across multiple benchmarks in the zero-shot setting, achieving 0.54 LLM queries per tweet while significantly enhancing performance. Our CoVer offers a more practical solution for LLM deploying for social media stance detection.

Keywords: Small and Large Language Model · Collaborative Interference · Stance Detection

1 Introduction

Stance Detection (SD) is a powerful tool to reveal public viewpoints across a variety of social events. It has many important applications in social research

* Min Liu is the corresponding author. This research was supported by the National Key Research and Development Program of China (Grant No. 2021YFB2900102), Natural Science Foundation Program (Grant No. 62472410 and 62202449).

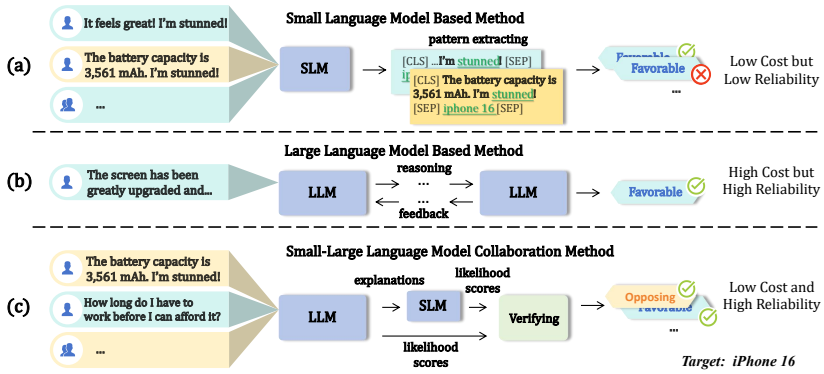

Fig. 1. Illustration of different stance detection methods for the target (*iPhone 16*) via Small Language Model (SLM), Large Language Model (LLM), and Small-Large Language Model (SLLM) Collaboration. (a) SLM-based method relies on pattern extraction, achieving low computational consumption but often struggling with background knowledge understanding. (b) LLM-based method leverages LLMs' reasoning capabilities for reliable stance detection but at a high computational cost. (c) Small-Large Language Model (SLLM) Collaboration method combines the strengths of both SLM and LLM to balance computational consumption and model performance, where LLM provides advanced reasoning, and SLM performs verification.

[1,9,12,14] including sentiment analysis, social media monitoring and rumor detection. In stance detection, each text is annotated with a stance label (Favor, Against, or None) toward a specific target. Due to the informal and ambiguous nature of expressions with heterogeneous user knowledge in those vast social media tweets, it poses challenges to efficient large-scale stance detection.

According to the computational scale of backbone model, stance detection methods can be categorized as Small Language Model Based (SLM-based) and Large Language Model Based (LLM-based) methods [5]. SLM-based methods [2,11,12,13] utilize the classifier to extract patterns from texts for stance detection. After task-specific training, SLM-based methods perform well on specific domain data, making them suitable for efficient processing [19]. However, their reliance on predefined patterns and specific keywords [10,12] limits their capability to generalize, dealing with implicit or subtle stances, as shown in Fig.1(a). In contrast, LLM-based methods[6,8,7,19] utilize general LLMs for stance detection, leveraging their reasoning and contextual understanding capabilities to comprehend diverse expression styles and knowledge in social media tweets. Despite the powerful strengths of LLMs, their stance detection performance still requires logical consistency verification to address issues such as hallucinations and outdated information, which can lead to inconsistencies between the LLM's reasoning and stance likelihood estimation, as shown in Fig.1(b).

Some recent studies have highlighted the importance of ensuring logical consistency between the LLM's stance reasoning and estimation for effective stance detection. These studies employ techniques such as multi-agent systems [6] and chain-of-thought [20,21] to address inconsistencies through iterative use of the LLM. However, these methods overlook that stance detection is a time-sensitive

task for large-scale data analysis, requiring both efficiency and accuracy. Repeatedly invoking LLM for a single short tweet is clearly cost-prohibitive.

To reduce redundant LLM utilization spent on calibrating logical inconsistencies, we innovatively put forward the insight of Small-Large Language Model (SLLM) Collaboration method, where the SLM collaborates with the LLM to ensure logical consistency in reasoning and stance likelihood estimation. As shown in Fig.1(c), consider the tweet "*The battery capacity is 3,561 mAh. I'm stunned!*", regarding the target "*iPhone 16*". The LLM recognizes that "*stunned*" conveys surprise but interprets it critically to low battery capacity, thus assigning an "*Against*" stance. For the LLM, its strength lies in its advanced reasoning and contextual understanding, which allows the LLM to explain background cues, making nuanced stance predictions even when sentiments are implicitly expressed. Then, the SLM is used to check that LLM's interpretation aligns with explicit indicators in the tweet, ensuring consistency between the reasoning and stance likelihood estimation. For the SLM, its strength lies in its ability to quickly recognize explicit patterns, which enables SLM to verify the consistency of LLM's predictions by checking for alignment with straightforward cues.

Building on these insights, we propose the ***Co****llaborative Stance Detection via Small-Large Language Model Consistency* ***Ver****ification* (**CoVer**) framework, which combines the LLM's reasoning capabilities with the SLM's verification efficiency. CoVer first reconstructs the context of tweets through knowledge augmentation and irrelevant context filtering, ensuring clear and unbiased stance reasoning. It then processes texts batch-by-batch, feeding each batch into the LLM for simultaneous reasoning, allowing for efficient context reuse. Finally, CoVer employs the SLM to verify the logical consistency of the LLM's reasoning for stance classification. For texts exhibiting repeated low consistency, CoVer performs consistency-weighted aggregation of likelihood scores for final classification. We perform extensive Zero-Shot Stance Detection (ZSSD) experiments on classic benchmarks, including SemEval-2016, VAST, and P-Stance. Experimental results show that CoVer outperforms state-of-the-art methods with only 0.54 LLM queries per text using GPT-4o-mini, highlighting its performance and resource efficiency. The major contributions of our study are as follows:

- We introduce CoVer, a Small-Large Language Model collaboration framework that utilizes the strengths of the LLM for reasoning and minimizes unnecessary re-queries of the LLM using the SLM, achieving the balancing of computational consumption and model performance.
- To further improve LLM utilization, we employ a batch-by-batch LLM reasoning approach in CoVer, which uses the LLM to process multiple texts with a single LLM query. Experiments indicate that CoVer not only reduces query overhead but also enhances performance through efficient context reuse by leveraging shared contextual cues across these texts.
- Our CoVer demonstrates significant performance improvements over several state-of-the-art methods across three classic benchmarks with only 0.54 LLM queries per tweet on zero-shot stance detection.

2 Problem Statement

Zero-Shot Stance Detection (ZSSD) [1,9,12,14] is defined as the task of classifying the stance (*Favor*, *Against*, *None*) expressed in a given tweet towards a specific target without providing any specific training data or reference samples about the target. In our study, for a given raw tweet x_{raw} toward target t, we pre-process it as the augmented tweet x, and develop a Large Language Model (LLM) and Small Language Model (SLM) collaboration framework to classifying the stance label y for x in zero-shot setting.

3 Methodology

In this section, we introduce our ***Co**llaborative Stance Detection via Small-Large Language Model Consistency **Ver**ification* (**CoVer**) framework, which combines the strengths of LLM and SLM to achieve balanced computational consumption and model performance in stance detection tasks. The overall structure of CoVer is shown in Fig.2 and the workflow of our CoVer is shown in Algorithm 1.

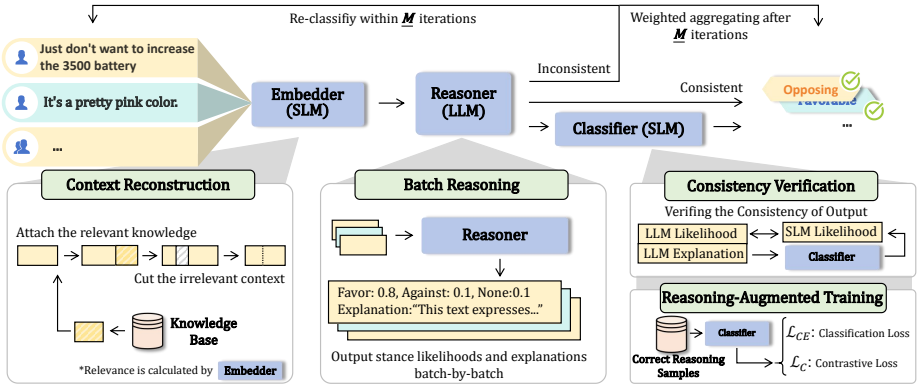

Fig. 2. The overall structure of our CoVer. Specifically, in Context Reconstruction (§3.1), CoVer first uses SLM to preprocess the input tweets by filtering out irrelevant contextual information and incorporating relevant knowledge to obtain the context-augmented tweets. Then, in batch reasoning (§3.2), CoVer uses LLM to perform stance reasoning and estimate the corresponding stance likelihood on a batch of tweets. Finally, in consistency verification (§3.3), CoVer uses SLM to verify the alignment between the LLM's reasoning and stance predictions. Those tweets with repeatedly low consistency will be classified through the consistency-weighted aggregation of LLMs' prior predictions. The SLM classifier is trained (§3.4) on LLM batch reasoning data.

3.1 Context Reconstruction

To ensure effective reasoning, it's crucial to optimize the input context for the LLM, especially when processing multiple tweets within a limited context window. Therefore, we employ Context Reconstruction to attach external knowledge and then filter out non-relevant content to ensure LLMs' unbiased reasoning.

Knowledge Augmentation To identify the stance of those tweets with implicit expressing, we introduce external knowledge to enrich the context.

Given the external knowledge base $\mathcal{K} = \{(k_0, d_0), (k_1, d_1)...\}$, where k_i is the knowledge entity, and d_i refers to the description associated with knowledge k_i, we employ entity linking match the knowledge. For those matched knowledge entries and corresponding descriptions, we concatenate them with the raw tweet x_{raw} to create an knowledge augmented tweet x_{k}:

$$x_{\text{k}} = x_{\text{raw}} \oplus \mathcal{D}, \mathcal{D} = \{k_i \oplus d_i \mid (k_i, d_i) \in \mathcal{K} \text{ and } k_i \in x_{\text{raw}}\}, \tag{1}$$

where $\oplus$ denotes the string concatenation, $\mathcal{E}$ is the set of matched knowledge via entity linking.

Sentence Filtering To determine a sentence or the knowledge weather contributes to externalizing the stance expression of the tweet, we measure its impact through **Stance Entropy (SE)**. A lower entropy for tweet x indicates more discriminative stance labels, suggesting the processed text effectively externalizes the stance. Entropy for tweet x is calculated as:

$$\text{SE}_x = -\sum_{y \in \mathcal{Y}} \hat{P}(y|x,t) \log \hat{P}(y|x,t), \tag{2}$$

where $\mathcal{Y}$ is the stance label set {Favor, Against, None}. A lower SE implies the tweet better externalizes its stance expression, as the stance likelihood is more concentrated on a specific stance.

To refine x_{k}, we split the knowledge augmented tweet x_{k} into the sentence set $X_{\text{k}} = \{s_1, s_2, \dots\}$ based on stance entropy. Our goal of sentence filtering is defined as:

$$X = \arg\max_{X \subseteq X_{\text{k}}} \text{SE}_x, \quad x = \bigoplus_{s_i \in X} s_i, \tag{3}$$

where x is the refined tweet concatenated from the optimal subset of sentences X that maximizes SE_x. We filter the irrelevant sentences according to the change in stance variance after the removal of each sentence. Specifically:

- **Redundant Sentence**: If removing a sentence s_i results in an obvious decrease in stance entropy $\text{SE}_{x \setminus s_i}$, i.e., $\mathbf{SE}_x \geq \mathbf{SE}_{x \setminus s_i}$, this suggests that s_i is redundant or has minimal impact on clarifying the stance. Thus, s_i is excluded from the tweet.
- **Relevant Sentence**: If removing a sentence s_i leads to a significant increase in stance entropy $\text{SE}_{x \setminus s_i}$, i.e., $\mathbf{SE}_x < \mathbf{SE}_{x \setminus s_i}$, this indicates that s_i contributes meaningfully to the stance expression, and it is retained.

To calculate the SE_x for tweet x, we estimate stance likelihood $\hat{P}(y|x,t)$. Specifically, we construct the stance phrase $s_y(t)$ that clearly expresses stance toward the target t, e.g., "My stance toward {target} is "{stance}". Then, we use the semantics similarity between tweets x and the stance phrases to estimate the $\hat{P}(y|x,t)$ as:

Batch Reasoning Prompt

You are a stance detection assistant...
Texts and Targets:

```
{text_target_pairs}
```

... Please respond in the following JSON format:

```
{
    "results": [
        {
            "text_id": Text ID,
            "explanation": "A brief explanation",
            "favor_probability": probability (0 to 1),
            "neutral_probability": probability (0 to 1),
            "against_probability": probability (0 to 1)
        },
        ...
    ]
}
```

$$\hat{P}(y|x,t) = \frac{\exp(\text{sim}(h_x, h_{s_y}))}{\sum_{y' \in \mathcal{Y}} \exp(\text{sim}(h_x, h_{s_{y'}}))}, \tag{4}$$

where we use the embedder (SLM, e.g., BGE-M3 [4]) for text semantic representation, h_x is the embedding of the tweet x, h_{s_y} is the embedding of the stance phrase $s_y(t)$, and $\text{sim}(\cdot,\cdot)$ is the cosine similarity between two embeddings.

3.2 Batch Reasoning

To enhance the LLM's utilization of context in stance detection, we group a batch of tweets as LLM input and classify the stance by reasoning. By processing a batch of tweets together, LLM gains access to a broader context that helps it understand relations between tweets, especially when tweets share a common theme or topic. Furthermore, the shared context also enhances the robustness and consistency of LLM's predictions across similar stance expressions.

To conduct the batch reasoning, we guide LLM to predict the stance likelihood $P_{\text{LLM},i}$ and output corresponding explanation e_i for each tweet x_i in the text batch $\mathcal{B} = \{(x_0, t_0), (x_1, t_1), ...(x_B, t_B)\}$:

$$\{(P_{\text{LLM},i}, e_i)\}_{i=1}^{B} = \text{LLM}(\text{prompt} \oplus \mathcal{B}), \tag{5}$$

$$P_{\text{LLM},i} = P(y|x_i, t_i, \mathcal{B}), \tag{6}$$

where $P(y|x_i, t_i, \mathcal{B})$ is the conditional probability of stance output by LLM for tweet x_i with respect to target t_i and the context of text batch $\mathcal{B}$, the prompt is the task instruction for stance detection. Shared context improves the model's ability to maintain consistency across similar stance expressions.

3.3 Consistency Verification

However, due to the fact that cross-influence of contextual information can potentially lead LLM to mistakenly apply the context of one tweet to another, it is important to exclude such negative influence caused by the shared context.

To verify LLM's predictions, we use the SLM as a third-party model to observe only the explanation for stance classification and compare the stance entropy of the prediction distribution before and after LLM reasoning. Specifically, for the corresponding explanation e_i generated by LLM for the tweet x_i from formula (5), it serves as the input to SLM:

$$P_{\text{SLM},i} = P(y|e_i, t_i) = \text{SLM}([CLS]e_i[SEP]t_i[SEP]), \tag{7}$$

where $P_{\text{SLM},i} = P(y|e_i, t_i)$ is the stance likelihood produced by the SLM based solely on the explanation e_i for tweet x_i toward target t_i.

Then, we calculate the stance entropy of explanation e_i and tweet x_i, denoted as SE_{e_i} and SE_{x_i}, and stance likelihood similarity between SLM and LLM using the cosine similarity, denoted as $\text{sim}(P_{\text{LLM},i}, P_{\text{SLM},i})$. Based on the above variables, our consistency verification mechanism is as follows:

- **Invalid Prediction:** If LLM reasoning can not expose the stance, i.e., $SE_{e_i} > SE_{x_i}$,the prediction and corresponding explanation generated by LLM for x_i is invalid. x_i will be re-classified.
- **Valid Prediction:** If LLM reasoning exposes the stance and the prediction distribution from SLM and LLM is consistent, i.e., $SE_{e_i} \leq SE_{x_i}$ **and** $\textbf{sim}(P_{\textbf{LLM},i}, P_{\textbf{SLM},i}) \geq \delta$, the prediction and corresponding explanation generated by LLM for x_i is valid. $P_{\text{LLM},i}$ will be used as the predicted result for tweet x_i.
- **Referable Prediction:** If LLM reasoning exposes the stance but the prediction distribution from SLM and LLM is inconsistent, i.e., $SE_{e_i} > SE_{x_i}$ **but** $\textbf{sim}(P_{\textbf{LLM},i}, P_{\textbf{SLM},i}) < \delta$, the prediction and corresponding explanation generated by LLM for x_i is referable. x_i will be re-classified and $P_{\text{LLM},i}$ will be used for weighted-aggregation as the final prediction if the stance of x_i still can not be correctly predicted after M round classifying.

After M rounds of re-classification, for those tweets only with invalid predictions, a stronger LLM will be used for classifying. Those tweets only with referable predictions will be predicted by consistency-weighted aggregation as:

$$P_{\text{Agg}}(y|x_i, t_i) = \sum_{j=1}^{M'} w_j \cdot P^{(j)}_{\text{LLM},i}, \tag{8}$$

where $w_j = \text{sim}(P^{(j)}_{\text{LLM},i}, P^{(j)}_{\text{SLM},i})$ is the weight assigned to the j-th round prediction. This weighted aggregation ensures a robust decision for those difficult classifying tweets. M' is the number of referable predictions for tweet x_i.

3.4 Reasoning-Augmented Training

To ensure the SLM classifier in consistency verification (§3.3) is capable of verifying the correctness of LLM reasoning, we fine-tune a BERT model [13] as the classifier. To ensure that the classifier learns the correct reasoning patterns, we trained it on data collected from LLMs' correct reasoning explanations. We introduce the multi-task learning framework combining the cross-entropy and the contrastive loss [10] as:

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \cdot \mathcal{L}_{C}, \tag{9}$$

$$\mathcal{L}_{CE} = -\frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{j=1}^{|\mathcal{Y}|} y_{i,j} \log(P_{i,j}), \tag{10}$$

$$\mathcal{L}_{C} = -\frac{1}{|\mathcal{B}|} \sum_{x_i \in \mathcal{B}} \ell_c(\mathbf{h}_{x_i}), \tag{11}$$

$$\ell_c(\mathbf{h}_{x_i}) = \log \frac{\sum_{(x_i, x_j^+) \in \mathcal{P}_i} \exp(\text{sim}(\mathbf{h}_{x_i}, \mathbf{h}_{x_j}^+)/\tau_s)}{\sum_{x_j \in \mathcal{B} \setminus x_i} \exp(\text{sim}(\mathbf{h}_{x_i}, \mathbf{h}_{x_j})/\tau_s)}, \tag{12}$$

where λ is weight hyperparameter, τ_s is temperature hyperparameter, $\mathbf{h}_{(\cdot)}$ is the embedding of tweet output by SLM. $\mathcal{B} = \{x_0, x_1, ...\}$ is mini-batch training data, and $\mathcal{B} \setminus x_i$ is $\mathcal{B}$ excluding sample x_i. $\mathcal{P}_i = \{(x_i, {x_0}^+), (x_i, {x_1}^+), ...\}$ is the set of positive pairs for x_i, which consists of samples with the same label in mini-batch.

3.5 Workflow of CoVer

To provide a clear understanding of the workflow of CoVer, we present a detailed pseudocode in Algorithm 1, which focuses on the stages of reasoning and stance classification of CoVer.

4 Experiments

4.1 Experiment Settings

Datasets To demonstrate the effectiveness of CoVer, we perform experiments of zero-shot stance detection on three benchmarks: **Sem16 (SemEval-2016)** [14], **P-stance** [9], and **VAST** [1]. For Sem16 and P-stance, we use the leave-one-target-out evaluation setup. For the VAST dataset, we use their original zero-shot dataset settings. We adhere to standard train, validation, and test splits in alignment with previous studies [6,7,8].

Evaluation Metrics We adopt the typical metric employed in stance detection [14,15,18] to evaluate the effectiveness of different methods, denoted as F_{AVG}. For sem16 and P-stance, F_{AVG} is computed by averaging the F1-scores of the "Favor" and "Against" categories. For VAST, F_{AVG} is computed by averaging the F1-scores of "Pro", "Con" and "Neutral" categories. To evaluate different methods' utilization efficiency of LLMs, we use Q_{AVG} to measure the average query count required for one sample stance detection.

Algorithm 1: The workflow of CoVer

Input: Text Dataset $\mathcal{X} = \{(x_i, t_i)\}_{i=1}^{N}$
Output: Stance labels $\mathcal{L} = \{l_i\}_{i=1}^{N}$
while *round* $< M$ **do**
  Divide $\mathcal{X}$ into batches $\{\mathcal{B}_1, \mathcal{B}_2, \dots\}$;
  Context Reconstruction(§3.1) Refine each tweet within text batch $\mathcal{B}$;
  Batch Reasoning(§3.2) $\{(p_{\text{LLM},j}, \text{reason}_j)\}_{j=1}^{B} \leftarrow \text{LLM}(\text{prompt} \oplus \mathcal{B})$;
  Consistency Verification(§3.3)
  foreach $(p_{LLM,j}, reason_j)$ **do**
    $s_j \leftarrow \text{sim}(p_{\text{SLM}}, p_{\text{LLM}})$, where $p_{\text{SLM},j} \leftarrow \text{SLM}(\text{reasons}_j)$;
    if s_j *exceeds threshold* **then**
      Add $l_j = \arg\max(p_{\text{LLM},j})$ to $\mathcal{L}$;
    end
    else
      Retain for re-detect;
    end
  end
  Update $\mathcal{X}$ by removing classified instances, $round+ = 1$;
end
Apply weighted aggregation on retained instances to finalize $\mathcal{L}$;
return $\mathcal{L}$

Experimental Setups In CoVer[5], the SLM embedder employs the general-purpose embedding model BGE-M3 [4]. Knowledge is derived from the description of concepts in the training data, generated using GPT-4o. CoVer performs batch reasoning iteratively with LLMs across three rounds. The batch sizes for each round are 8, 4, and 1, utilizing gpt-4o-mini-2024-07-18 as LLM with a model temperature set to 0.1 for all iterations, GPT-4o is used for the final invalid prediction. Test data batches are randomly shuffled. The classifier SLM in CoVer uses a Roberta [13], trained on samples predicted correctly by the LLM from the Sem16, P-Stance, and VAST training sets. The contrastive loss weight is set to 0.1. The consistency threshold between the LLM reasoner and SLM classifier is set to 0.9. For SLM training, λ is 0.1, τ_s is 0.05, batch size is 32.

Baseline Methods We provide an overview of the baseline methods for comparison in our experiments, including Small Language Model Based Methods: 1) **BERT-GCN** [12] leverages commonsense knowledge from ConceptNet to improve the model's generalization. 2) **TOAD** [2] uses adversarial learning to improve zero-shot stance detection, enabling effective stance classification on unseen targets. 3) **JointCL** [10] uses joint contrastive learning framework. 4) **PT-HCL** [11] leverages hierarchical contrastive learning to distinguish between target-invariant and target-specific stance features. 5) **TGA-Net** [11] applies topic-grouped attention to capture relationships between targets. 6) **TarBK-BERT** [22] leverages targeted background knowledge from Wikipedia to improve performance. Large Language Model Based Methods: 7) **KASD** [7] leverages episodic knowledge from Wikipedia and discourse knowledge for knowledge

[5] Our code is available at `https://github.com/qzqdz/CoVer`.

augmentation. 8) **COLA** [6] uses a multi-agent framework to debate the stance of tweets. 9) **LC-CoT** [20] employs the structured chain-of-thought approach for stance detection. 10) **Task-Des** [8] uses task-related descriptions for stance detection. 11) **Task-CoT-Demo** [8] uses the task description with 4-shot chain-of-thought demonstration.

4.2 Experimental Results

We aim to answer the following research questions (RQs) by conducting a series of experiments:

- **RQ1:** Does CoVer demonstrate superior effectiveness and adaptability compared to existing state-of-the-art stance detection methods?
- **RQ2:** If CoVer outperforms existing methods, what mechanisms contribute to its success?
- **RQ3:** Given that CoVer employs multiple re-generations strategy for those samples with low consistency, does this imply lower efficiency compared to other LLM-based methods?

Baseline Comparison (RQ1) To answer RQ1, the baseline comparison experiment is conducted. As evidenced in Table 1 and Table 2, CoVer attains a comparable performance to the baselines on cross-target datasets. Specifically, CoVer achieves the best performance on sem16 and P-stance by outperforming the top existing methods with 1.98% and 2.44% on average, and outperforming all large language model based methods on VAST, showcasing robust effectiveness and adaptability across datasets.

We observe a clear performance gap between small and large language model based methods. LLMs utilize their internal commonsense and background knowledge for effective stance inference. In contrast, SLMs depend on heuristic training and explicit background knowledge modeling, limiting their generalization in scenarios with imbalanced targets, such as CC, where no SLM-based method exceeds 41%, and in low-resource settings, such as Sem16, where only KASD-BERT's average performance surpasses that of the weakest LLM-based method, KASD-LLaMA-2. Furthermore, we also observe that LLMs cannot fully utilize their capabilities without consistency verification, such as GPT-3.5-Turbo-CoT-Demo outperforms GPT-3.5-Turbo-Task-Des 12.06% on Sem16 and 19.93% on VAST. Therefore, CoVer enhances consistency verification utilizing SLM, which is more efficient and effective than relying solely on LLMs for verification.

Ablation Study (RQ2) To answer RQ2, the contributory of every component in CoVer is investigated by ablation study as shown in Table 3. The ablation settings and analysis are as follows:

Effectiveness of Consistency Verification (Ver.) Ver. plays a crucial role in enhancing CoVer's overall performance by ensuring reasoning consistency. To evaluate it, we remove Ver. from CoVer for testing, (denoted as w/o Ver.).

Table 1. Zero-Shot Stance Detection Experiments on Sem16 and VAST datasets. The best results are in **bold** and the second-best results are in underline. Results with * denote that CoVer significantly outperforms baselines with the p-value < 0.05.

Model	Sem16 (%)						VAST (%)
	HC	FM	LA	A	CC	Avg	All
Small Language Model Based Methods							
BERT-GCN	50.00	44.30	44.20	53.60	35.50	48.03	68.60
TOAD	51.20	54.10	46.20	46.10	30.90	49.40	41.00
JointCL	54.80	53.80	49.50	54.50	39.70	53.15	72.30
PT-HCL	54.50	54.60	50.90	56.50	38.90	54.13	71.60
TGA-Net	49.30	46.60	45.20	52.70	36.60	48.45	66.60
TarBK-BERT	55.10	53.80	48.70	56.20	39.50	53.45	73.60
KASD-BERT	64.78	57.13	51.63	55.97	40.11	57.38	**76.82**
Large Language Model Based Methods							
KASD-LLaMA-2	77.70	65.57	57.07	39.55	39.55	55.89	43.42
LLaMA-2-Task-Des	73.79	71.03	66.00	60.44	61.91	66.63	68.54
LLaMA-2-CoT-Demo	72.09	**73.83**	66.50	57.58	65.11	67.02	67.28
GPT-3.5-Turbo-Task-Des	72.70	71.71	67.89	28.87	59.36	60.11	50.21
GPT-3.5-Turbo-CoT-Demo	78.69	73.22	**72.24**	65.15	71.54	72.17	70.14
KASD-ChatGPT	80.32	70.41	62.71	63.95	55.83	66.64	67.03
COLA	75.90	69.10	71.00	62.30	64.00	68.46	73.40
LC-CoT	**82.90**	70.40	63.20	-	-	-	72.50
Small-Large Language Model Based Method							
CoVer (ours)	81.17*	73.35	72.01*	**70.40***	**73.81**	**74.15***	74.79

Experimental results indicate that without Ver., CoVer's F_{AVG} significantly decreases by 5.00% on Sem16, 6.90% on VAST and 4.85% on P-stance, requiring less LLM queries Q_{AVG}. Without Ver., the LLM performs reasoning without verification, potentially leading to more biased outputs and negatively impacting overall performance. This phenomenon further highlights the importance of ensuring the consistency of reasoning. Different from existing LLM self-verification approaches, Ver. ensure the LLM's reasoning consistency via SLM with fewer (0.54 on average) LLM queries per tweet.

Effectiveness of Contextual Reconstruction (Ctx.) Ctx. Largely ensures CoVer's performance. To evaluate it, we remove the Ctx. from CoVer for testing (denoted as w/o Ctx.). Experimental results show that without Ctx., the CoVer's F_{AVG} decreases by 3.62% on Sem16, 4.35% on VAST, and 1.43% on P-stance. Additionally, the higher Q_{AVG} indicates that the lack of context augmentation also causes the inefficiency of the overall methods. This phenomenon suggests that the clearer context allows CoVer to capture implicit reasons and key information in tweets, thereby ensuring LLM to generate the consistent reasoning. By reconstructing context, CoVer can more effectively and efficiently reason the stance for those ambiguous tweets and lengthy tweets.

Effectiveness of Batch Reasoning (Bat.) Bat. plays a crucial role in improving the LLM utilization of CoVer. To evaluate it, we remove the batch reasoning for testing (denoted as w/o Bat.). Experimental results show that without Bat., the CoVer's F_{AVG} lightly decreases by 3.68% on Sem16, while Q_{AVG} increases significantly by 1.96 on Sem16, 1.29 on VAST and 1.34 on P-stance.

Table 2. Zero-Shot Stance Detection Experiments on the P-stance dataset. The best results are in **bold** and the second-best results are in underline. Results with * denote that CoVer significantly outperforms baselines with the p-value < 0.05.

Method	P-stance (%)			
	Biden	Sanders	Trump	Avg
TarBK-BERT	75.49	70.45	65.80	70.58
KASD-BERT	79.04	75.09	70.90	74.09
KASD-LLaMA-2	75.28	74.09	69.29	72.87
KASD-ChatGPT	83.12	82.14	82.04	82.28
COLA	83.60	79.66	84.31	82.52
CoVer (ours)	**85.86***	**82.63**	**86.40***	**84.96***

Table 3. Experimental results of Ablation Study on Sem16, P-stance and VAST. The best result is highlighted in **bold**. The second best result is highlighted in underline.

Variants	Sem16		VAST		P-stance	
	$F_{\text{AVG}}(\uparrow,\%)$	$Q_{\text{AVG}}(\downarrow)$	$F_{\text{AVG}}(\uparrow,\%)$	$Q_{\text{AVG}}(\downarrow)$	$F_{\text{AVG}}(\uparrow,\%)$	$Q_{\text{AVG}}(\downarrow)$
CoVer	**74.15**	0.53	74.79	0.54	84.96	0.54
w/o Ver.	$69.15_{\downarrow 5.00}$	**0.35**	$67.89_{\downarrow 6.90}$	**0.26**	$80.11_{\downarrow 4.85}$	**0.21**
w/o Ctx.	$70.53_{\downarrow 3.62}$	$0.99_{0.46\uparrow}$	$70.44_{\downarrow 4.35}$	$0.67_{0.13\uparrow}$	$83.53_{\downarrow 1.43}$	$0.98_{0.44\uparrow}$
w/o Bat.	$70.47_{\downarrow 3.68}$	$2.49_{1.96\uparrow}$	**75.18**	$1.83_{1.29\uparrow}$	**85.20**	$1.88_{1.34\uparrow}$

The dramatic increase in Q_{AVG} with bat. further highlights its importance in reducing redundant LLM utilization. Furthermore, instead of introducing stance biases or misclassifications, such shared context in batch processing also could enhance the robustness of LLM's internal stance comprehension criteria under some conditions. As evidenced by CoVer's improved F_{AVG} on Sem16, the performance enhancing by batch reasoning indicates that its potential effectiveness for minimizing LLM's reasoning biases.

Efficiency Comparison (RQ3) To answer RQ3, we selected several LLM-based methods for a comparison of model performance and LLM utilization on Sem16, as shown in Table 4. We can observe that CoVer achieves the highest F_{AVG} 74.15% with the lowest average query count (Q_{AVG}) 0.53 with less complicated prompt tactics. The comparison results demonstrate that CoVer achieves high performance with less LLM utilization by combining LLM batch reasoning with SLM consistency verification. This efficiency is attributed to CoVer's SLM and LLM collaboration mechanism, which leverages the strengths of the LLM for reasoning and uses SLM to reduce redundant queries to the LLM.

Case Study To illustrate CoVer's consistency verification of LLM reasoning to ensure correct predictions, we conduct the case study shown in Figure 3. In this case, the tweet implies a critique of alimony but does not explicitly connect this critique to the "*Feminist Movement*", making the stance challenging to classify with certainty. Both reasoner 1 and reasoner 3 predict a "Neutral" stance, with moderate consistency scores (0.8341 and 0.8119, respectively), interpreting the

Table 4. Prompt efficiency comparison across different LLM-based methods including DQA (Direct Question-Answering [21]), StSQA (Step-by-Step Question-Answering [21]), KASD-ChatGPT and COLA. The more "✓" a method has, the less efficient its LLM utilization is. Single-C: Single text Classification. T-Demo: Task Demonstration. K-Aug: Knowledge Augmentation. M-Round: Multiple Round.

Method	Prompt Tactics					Sem16	
	Single-C	T-Demo	K-Aug	CoT	M-Round	$F_{\text{AVG}}(\uparrow, \%)$	$Q_{\text{AVG}}(\downarrow)$
CoVer (ours)	✗	✓	✓	✗	✓	**74.15**	**0.53**±0.29
DQA	✓	✗	✗	✗	✗	$48.22_{\downarrow 25.93}$	$\underline{1.00}$ $_{0.47\uparrow}$
StSQA	✓	✓	✗	✓	✓	$49.35_{\downarrow 24.80}$	3.00 $_{2.47\uparrow}$
KASD-ChatGPT	✓	✓	✓	✓	✓	$\underline{68.46}_{\downarrow 5.69}$	3.00 $_{2.47\uparrow}$
COLA	✓	✓	✓	✓	✓	$66.64_{\downarrow 7.51}$	6.00 $_{5.47\uparrow}$

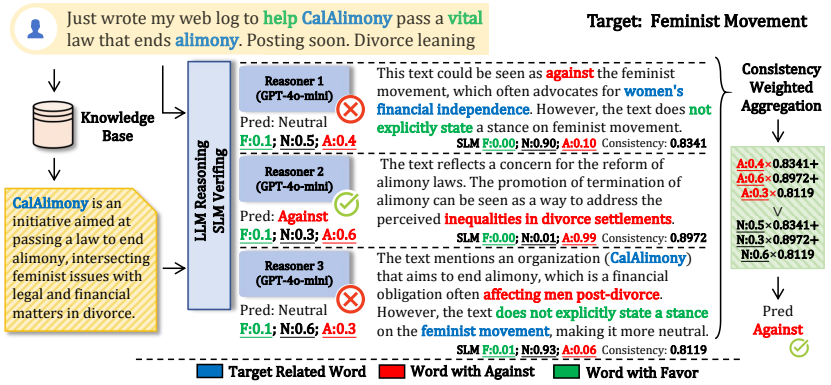

Fig. 3. Case study of CoVer, where the tweet only mentions "*CalAlimony*" in support of ending alimony, which seems neutral but implies a critique relevant to "*Feminist Movement*" indirectly. CoVer uses weighted aggregation to verify consistency across different LLM outputs, leveraging the SLM to ensuring the correct stance prediction.

text as lacking an explicit critical stance towards "*Feminist Movement*". Their explanations highlight that, while the text discusses alimony reform, it does not directly oppose "*Feminist Movement*". In contrast, reasoner 2 predicts an "Against" stance with a higher consistency score (0.8972), suggesting it interprets the text as implicitly critical of alimony, aligning with an opposition stance towards the "*Feminist Movement*".

Through weighted aggregation, CoVer assigns higher weight to reasoner 2 due to its higher consistency score, resulting in an"Against" stance as the final prediction. This case demonstrates CoVer's ability to reconcile differing model outputs through weighted aggregation, achieving accurate stance classification even when model interpretations vary.

4.3 Discussion of CoVer

A fundamental component of CoVer is using batch reasoning to improve model efficiency. Intuitively, such shared context could introduce negative cross-influence

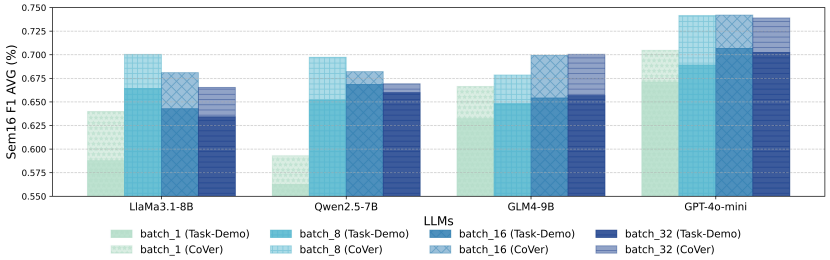

Fig. 4. Comparative analysis of F_{AVG} acorss different LLMs and batch settings on Sem16, demonstrating the scalability of CoVer's batch reasoning. Results demonstrate that increasing batch sizes ($1 \rightarrow 32$) does not necessarily degrade model performance. CoVer effectively leverages batch reasoning to ensure the robustness of stance detection across different LLMs.

between tweets, potentially causing bias in stance predictions. However, our ablation study in §4.2 has shown that increasing batch size does not necessarily degrade model performance. This phenomenon warrants further investigation into the correlation between batch size scaling and LLM performance.

We conduct experiments across different LLMs with varying batch sizes on the Sem16 dataset. We compare the Cover with the Task-Demo baseline, whose task instruction consistent with CoVer, across batch sizes (1, 8, 16, 32) on four advanced LLMs (LLAMA-3.1-8B[6], Qwen2.5-7B[7], GLM4-9B[8], GPT-4o-mini[9]). As shown in Fig.4, experimental results indicate that: 1) Single-sample processing does not achieve the best performance across different LLMs. Compared to single-sample processing, batch processing allows LLMs to simultaneously process multiple samples, which ensures the establishment of more robust pattern recognition and decision criteria. 2) Different LLMs demonstrate model-specific optimal batch sizes, e.g., LlaMa3.1-8B is 8, while Qwen2.5-7B and GPT-4o-mini is 16, GLM4-9B is 32. This can be attributed to their capability for long-sequence processing. 3) CoVer consistently improves LLM performance across batch sizes, e.g., CoVer achieves an improvement of over 5% on GPT-4o-mini. This suggests that the consistency verification and context reconstruction of CoVer can effectively remove the biases in LLM batch reasoning.

Our CoVer validates the feasibility of batch reasoning. Furthermore, through the collaboration between SLM and LLM, CoVer achieves a balanced trade-off between model effectiveness and computational efficiency, making it a practical solution for real-world applications.

5 Related work

Stance Detection via Knowledge-Augmentation To enhance the understanding and classification of a stance in a given text [7,12], many studies leverage

[6] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct
[7] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
[8] https://huggingface.co/THUDM/glm-4-9b-chat
[9] https://platform.openai.com/docs/models/gpt-4o-mini

external knowledge sources, such as knowledge graphs [12], structured databases [3], and external textual information [22] for knowledge augmentation. By incorporating external knowledge such as DBpedia [3] or ConceptNet [16], models can gain a deeper contextual understanding, particularly useful for identifying implicit stances or understanding domain-specific terminology. Additionally, recent studies [17] indicate that integrating factual and contextual knowledge can significantly enhance the model's ability to detect subtle or implicit stances, especially in scenarios with limited or biased training data.

In summary, knowledge augmentation has been proven by existing studies to be an effective strategy for enhancing stance classification. It addresses information insufficiency by providing context, resolving ambiguities, and identifying subtle relationships between the text and the target, which is especially effective in complex scenarios where direct textual information is limited.

Stance Detection via Reasoning Many studies [11,12,21] emphasize identifying stances in text through logical reasoning. These methods focus on analyzing arguments, causal relations, and implicit cues within the text to determine the stance, making them particularly effective in few-shot and zero-shot scenarios with complex arguments. Recently, some studies have combined LLMs with such strategies to generate reasoning chains for stance detection. Specifically, the Logically Consistent Chain-of-Thought (LC-CoT) [21] enhances zero-shot stance detection by evaluating external knowledge requirements, invoking APIs to retrieve background knowledge, and employing if-then logic templates to generate reasoning chains. The Collaborative Role-Infused LLM-based Agents (COLA) [6] sets up multi-role LLM agents (e.g., linguistic experts, domain specialists, social media experts) for multi-view analysis.

In summary, stance detection via reasoning effectively handles implicit meanings and multi-step reasoning contexts by logical reasoning, demonstrating significant advantages in few-shot and zero-shot scenarios.

6 Conclusion

In this study, we propose ***Co****llaborative Stance Detection via Small-Large Language Model Consistency* ***Ver****ification* (**CoVer**), which combines the strengths of LLM and SLM to balance the computational consumption and model performance. Specifically, to ensure unbiased stance reasoning, CoVer uses the context reconstruction module for knowledge augmentation and irrelevant context filtering. Then, to improve the utilization of LLM, CoVer introduces the batch reasoning module, allowing the LLM to process multiple tweets simultaneously. Finally, to ensure the correctness of stance classification, CoVer employs a consistency verification module with an SLM to align reasoning and stance predictions. For tweets that repeatedly show low consistency, CoVer classifies them using a consistency-weighted aggregation of the likelihood scores. Experimental results have indicated that CoVer demonstrates state-of-the-art performance across various benchmarks and reduces LLM queries to 0.54 per tweet, which offers a more practical solution for stance detection on social media.

References

1. Allaway, E., Mckeown, K.: Zero-shot stance detection: A dataset and model using generalized topic representations. In: EMNLP 2020. pp. 8913–8931 (2020)
2. Allaway, E., Srikanth, M., McKeown, K.: Adversarial learning for zero-shot stance detection on social media. arXiv preprint arXiv:2105.06603 (2021)
3. Auer, S.e.a.: Dbpedia: A nucleus for a web of open data. In: ISWC. pp. 722–735. Springer (2007)
4. Chen, J., Xiao, S., Zhang, P.e.a.: M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In: Findings of ACL 2024. pp. 2318–2335 (2024)
5. Cruickshank, I.J., Xian Ng, L.H.: Use of large language models for stance classification. arXiv e-prints pp. arXiv–2309 (2023)
6. Lan, X., Gao, C., Jin, D., Li, Y.: Stance detection with collaborative role-infused llm-based agents. In: AAAI 2024. vol. 18, pp. 891–903 (2024)
7. Li, A., Liang, B., Zhao, J., Zhang, B., Yang, M., Xu, R.: Stance detection on social media with background knowledge. In: EMNLP 2023. pp. 15703–15717 (2023)
8. Li, A., Zhao, J., Liang, B., Gui, L., Wang, H., Zeng, X., Wong, K.F., Xu, R.: Mitigating biases of large language models in stance detection with calibration. arXiv preprint arXiv:2402.14296 (2024)
9. Li, Y., Sosea, T., Sawant, A., Nair, A.J., Inkpen, D., Caragea, C.: P-Stance: A Large Dataset for Stance Detection in Political Domain. In: ACL-IJCNLP Findings. pp. 2355–2365 (2021)
10. Liang, B.e.a.: JointCL: Joint Contrastive Learning Framework for Zero-shot Stance Detection. In: ACL. pp. 81–91 (2022)
11. Liang, B.e.a.: Zero-shot stance detection via contrastive learning. In: WWW. pp. 2738–2747 (2022)
12. Liu, R., Lin, Z., Tan, Y., Wang, W.: Enhancing zero-shot and few-shot stance detection with commonsense knowledge graph. In: ACL-IJCNLP 2021. pp. 3152–3157 (2021)
13. Liu, Y.e.a.: RoBERTa. arXiv preprint (2019), arXiv:1907.11692
14. Mohammad, S.e.a.: SemEval-2016 Task 6: Detecting Stance in Tweets. In: SemEval. pp. 31–41 (2016)
15. Sobhani, P.e.a.: Dataset for Multi-target Stance Detection. In: EACL. pp. 551–557 (2017)
16. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: AAAI. vol. 31 (2017)
17. Wang, J., Wang, C., Tan, C., Huang, J., Gao, M.: Boosting in-context learning with factual knowledge. arXiv preprint arXiv:2309.14771 (2023)
18. Xu, R.e.a.: Overview of NLPCC Shared Task 4: Stance Detection in Chinese Microblogs. In: NLUIA, pp. 907–916. Springer (2016)
19. Zhang, B., Ding, D., Jing, L.: Future of Stance Detection Techniques Post ChatGPT. arXiv preprint (2022), arXiv:2212.14548
20. Zhang, B., Ding, D., Jing, L., Huang, H.: A logically consistent chain-of-thought approach for stance detection. arXiv preprint arXiv:2312.16054 (2023)
21. Zhang, B., Fu, X., Ding, D., Huang, H., Li, Y., Jing, L.: Investigating chain-of-thought with chatgpt for stance detection on social media. arXiv preprint arXiv:2304.03087 (2023)
22. Zhu, Q., Liang, B., Sun, J., Du, J., Zhou, L., Xu, R.: Enhancing zero-shot stance detection via targeted background knowledge. In: ACM SIGIR. pp. 2070–2075 (2022)