LLMs Can Also Do Well! Breaking Barriers in Semantic Role Labeling via Large Language Models

Xinxin Li[1,*], **Huiyao Chen**[1,*], **Chengjun Liu**[2], **Jing Li**[1]
Meishan Zhang[1,†], **Jun Yu**[3], **Min Zhang**[1]
[1]Institute of Computing and Intelligence, Harbin Institute of Technology (Shenzhen), China
[2]School of Electrical Engineering and Automation, Harbin Institute of Technology, China
[3]Computer Science Department, Harbin Institute of Technology (Shenzhen), China
{lixinx0714,chenhy1018,mason.zms}@gmail.com

Abstract

Semantic role labeling (SRL) is a crucial task of natural language processing (NLP). Although generative decoder-based large language models (LLMs) have achieved remarkable success across various NLP tasks, they still lag behind state-of-the-art encoder-decoder (BERT-like) models in SRL. In this work, we seek to bridge this gap by equipping LLMs for SRL with two mechanisms: (a) retrieval-augmented generation and (b) self-correction. The first mechanism enables LLMs to leverage external linguistic knowledge such as predicate and argument structure descriptions, while the second allows LLMs to identify and correct inconsistent SRL outputs. We conduct extensive experiments on three widely-used benchmarks of SRL (CPB1.0, CoNLL-2009, and CoNLL-2012). Results demonstrate that our method achieves state-of-the-art performance in both Chinese and English, marking the first successful application of LLMs to surpass encoder-decoder approaches in SRL.

1 Introduction

Semantic role labeling (SRL) is a fundamental task in natural language processing (NLP, Gildea and Jurafsky, 2000), aiming to analyze the semantic relationships between predicates and their corresponding arguments in a sentence (Pradhan et al., 2005; Lei et al., 2015; Chen et al., 2025). SRL is crucial for various NLP applications, including information extraction (Barnickel et al., 2009; Christensen et al., 2010, 2011; Evans and Orasan, 2019), question answering (Shen and Lapata, 2007; Berant et al., 2013; Yih et al., 2016), machine translation (Liu and Gildea, 2010; Shi et al., 2016; Marcheggiani et al., 2018), robot command parsing (Bastianelli et al., 2013, 2014; Thomason et al., 2020; Garg et al., 2020; Lu et al., 2021) and etc. Besides, we can also use SRL to enhance pretrained language models with structured-aware semantic information (Zhang et al., 2020; Xu et al., 2020).

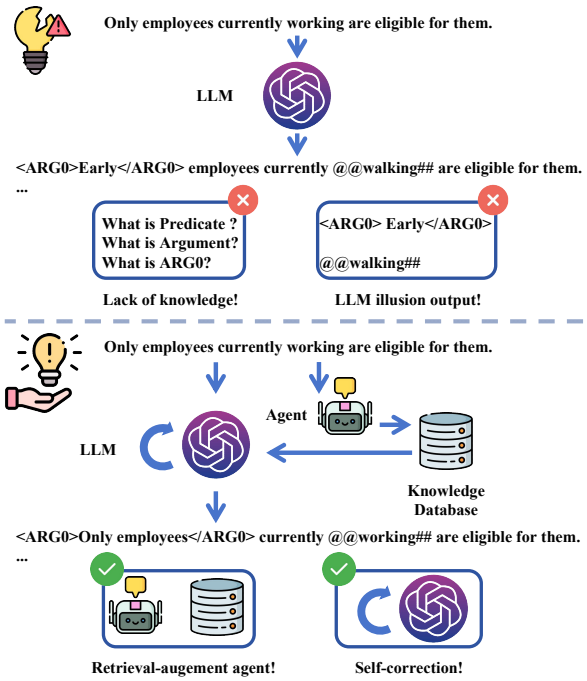


Figure 1: Challenges of direct LLM application in SRL and our solutions. <ARG0> </ARG0> denotes the role, with the enclosed tokens representing the argument, while @@## highlights the predicate.

Recently, the advent of large language models (LLMs) has reformulated NLP tasks with prompt-based generative solutions (Brown et al., 2020). Impressive performance has been achieved across a wide range of tasks (Achiam et al., 2023; Wang et al., 2024; Gu et al., 2025). The reason behind lies in the powerful capability of LLMs for general reasoning (Wei et al., 2022). For most NLP tasks, such as information extraction (Zhong et al., 2021), sentiment analysis (Radford et al., 2019), question answering (Khashabi et al., 2020), and (vallina) machine translation (Jiao et al., 2023), which require little specialized expertise, LLMs can handle them easily. Thus, it is reasonable to expect that

[*] Authors contributed equally.
[†] Corresponding author: Meishan Zhang.

Findings of the Association for Computational Linguistics: ACL 2025, pages 23162–23180

LLMs can perform well on most of these tasks.

For several NLP tasks, this is not the case. Unfortunately, SRL is one of them, as it requires strong explainability in linguistics (Sun et al., 2023; Chen et al., 2024; Cheng et al., 2024). As shown in Figure 1, the upper part illustrates a representative baseline of LLM performance in resolving SRL tasks. Without linguistic knowledge, what is the meaning of predicate-argument structures, `ARG0`∼`5`, and other related concepts? Clearly, this is not an easy question that can be answered through general reasoning alone. A solid background in linguistic knowledge is required to provide an explanation. Furthermore, the illusion problem in LLMs is another serious issue, as SRL results are highly dependent on the input data.

In this work, we propose two novel mechanisms to sufficiently enhance the expertise of LLMs in addressing the above two problems, respectively. First, we design a retrieval-augmented agent to enable LLMs to better understand predicates and their semantic arguments. We construct an external knowledge database based on the predicate-argument description guidance document from the original dataset. Second, to mitigate the illusion outputs of LLMs, we allow the LLM to verify its outputs by evaluating and correcting them autonomously, a process referred to as self-correction.

Concretely, we design a two-stage conversation-based architecture to perform SRL with LLMs: (1) predicate identification and (2) argument labeling, which are aligned with traditional BERT-like encoder-decoder systems. Based on this architecture, we integrate the aforementioned two mechanisms. LLM reasoning in the two stages is executed in an iterative manner. In addition, LLM parameters are also optimized to better suit our SRL task.

We conduct comprehensive experiments covering both Chinese (Zh) and English (En) on three widely-used benchmarks: Chinese Proposition Bank 1.0 (CPB1.0) (Xue, 2008), CoNLL-2009 (Hajič et al., 2009), and CoNLL-2012 (Pradhan et al., 2012). The results demonstrate that our approach achieves state-of-the-art performance, marking the first time an LLM-driven method has surpassed traditional approaches on the complete SRL task. This breakthrough highlights not only the effectiveness of leveraging LLMs through our framework but also the critical role of retrieval-augmented agents in addressing the inherent challenges of SRL.

In summary, the main contributions of this work can be summarized as follows:

- We introduce an LLM-driven approach to SRL that annotates predicate-argument triples step-by-step.
- We design a retrieval-augmented framework that enhances SRL performance by integrating external knowledge about predicates and their frame descriptions.
- We achieve state-of-the-art results on three benchmark datasets, demonstrating the superiority of our approach over traditional methods in complete SRL tasks.

Our code and prompt templates will be publicly available at github.com/fangfang123gh/LLM-SRL to facilitate future research.

2 Related Work

SRL has been explored through transition-based, graph-based, and generative modeling paradigms.

Transition-based and Graph-based Methods. Transition-based methods construct SRL structures incrementally via a sequence of actions (Fernández-González and Gómez-Rodríguez, 2020; Fei et al., 2021b), while graph-based methods represent predicate-argument structures as graphs, enabling unified modeling of syntactic and semantic dependencies (Marcheggiani and Titov, 2017; Li et al., 2018). These approaches have achieved strong results, particularly when incorporating syntactic features or external linguistic resources. Recent advances include iterative refinement strategies that progressively improve SRL structures through multiple passes (Lyu et al., 2019), demonstrating the effectiveness of iterative processing in complex semantic parsing tasks.

Generative Approaches. Generative methods have shown promising potential by modeling the joint probability distribution of inputs and outputs. Early sequential models like Hidden Markov Models (Thompson et al., 2004) laid the foundation, while Yuret et al. (2008) captured inter-stage interactions through sophisticated generative frameworks. The emergence of sequence-to-sequence models (Daza and Frank, 2018) marked a paradigm shift, reformulating SRL as a unified generation task encompassing predicate sense disambiguation, argument identification, and classification. Blloshmi et al. (2021) further advanced this direction with an end-to-end generative framework, achieving excellent performance on both dependency-based and span-based SRL tasks.

LLM-based Methods. Recently, LLMs have

opened new possibilities for SRL through their strong reasoning capabilities. Sun et al. (2023) demonstrated the feasibility of using ChatGPT for argument labeling given predicates, while Cheng et al. (2024) systematically analyzed the ability of LLMs to capture structured semantics. However, these studies revealed some challenges, including difficulties with predicate-argument structures, absence of explicit domain knowledge, and struggles with self-correction and consistency. These findings underscore the need for integrating external knowledge and iterative reasoning mechanisms to fully realize the potential of LLMs for SRL.

3 Method

In this section, we propose a retrieval-augmented framework for SRL, designed to address challenges such as predicate-argument complexity and the need for external knowledge. By reformulating SRL as a step-by-step conversational task, the framework integrates retrieval-enhanced agents to incorporate external knowledge and self-correction mechanisms to iteratively refine outputs. The rationale for unifying the tasks is explained in Appendix A. As illustrated in Figure 2, our framework employs a two-stage pipeline where each stage is augmented with retrieval-based knowledge and self-correction mechanisms. Below, we detail each component along with the mathematical formulations and algorithms involved. A detailed algorithm description and pseudocode can be found in Appendix F.

3.1 Preliminary

Given a sentence input $X = w_1, w_2, \ldots, w_n$, the goal of SRL is to identify all triples $(P, A, R) = \{(p_1, a_1, r_1), \ldots, (p_m, a_m, r_m)\}$, where p_i represents the predicate, a_i indicates the associated argument, and r_i denotes the semantic role assigned to the argument.

3.2 Predicate Identification

The first step in SRL is to identify potential predicates within a given sentence. To address the challenges of predicate recognition, such as ambiguity, we introduce a retrieval-augmented agent that leverages external databases to provide relevant contextual information about predicates.

Special tagging. To guide the LLM in predicate identification, we follow Sun et al. (2023) by inserting special tags @@ and ## around identified predicates. This results in sentences annotated with gold-standard predicates:

$$Y^p = \{w_1, \ldots, @@p_i\#\#, \ldots, w_n\},$$

where for $\forall p \in P$, $@@p\#\#$ is an element of Y^p.

Retrieval-augmented generation. To enhance predicate recognition, the retrieval-augmented agent generates a list of candidate predicates and retrieves their corresponding explanations. Each SRL dataset includes a guideline document with explicit explanations E_{p_i} for each predicate p_i. These documents are organized into a searchable knowledge database. Concretely, the agent follows a systematic process that begins with lemmatizing words to convert them into their base forms:

$$X_{\text{base}} = \text{Lemmatize}(w_1, w_2, \ldots, w_n).$$

Next, the agent iterates through each word in the sentence X_{base}, constructing a candidate predicate list $\hat{P} = \{\hat{p}_1, \ldots, \hat{p}_k\}$. For each candidate predicate, the agent retrieves explanations from the knowledge database. All candidate predicates and their explanations are incorporated as contextual information, culminating in the final prompt $\mathcal{D}^1$:

$$\mathcal{D}^1 = X + C^p + \{(\hat{p}_i, E_{\hat{p}_i}) | i = 1, 2, \ldots, k\},$$

where C^p is the predicate identification task context. Finally, we feed the prompt $\mathcal{D}^1$ to the LLM to obtain the final output $\widetilde{Y}^p$.

Training. During training, the cross-entropy objective is used for parameter optimization:

$$\mathcal{L}_{\text{pred}} = -\sum_{t=1}^{|Y^p|} \log P(Y_t^p | Y_{<t}^p, \mathcal{D}^1),$$

where $|Y^p|$ is the length of the target text Y^p, and $P(Y_t^p | Y_{<t}^p, \mathcal{D}^1)$ is the probability of generating the word Y_t^p given the input prompt $\mathcal{D}^1$ and the previously generated words $Y_{<t}^p$ at time step t.

3.3 Argument Labeling

After predicate identification, the next step is to assign semantic roles to the arguments associated with the identified predicates. This involves two subtasks: argument identification and role classification, which are performed simultaneously to minimize error propagation.

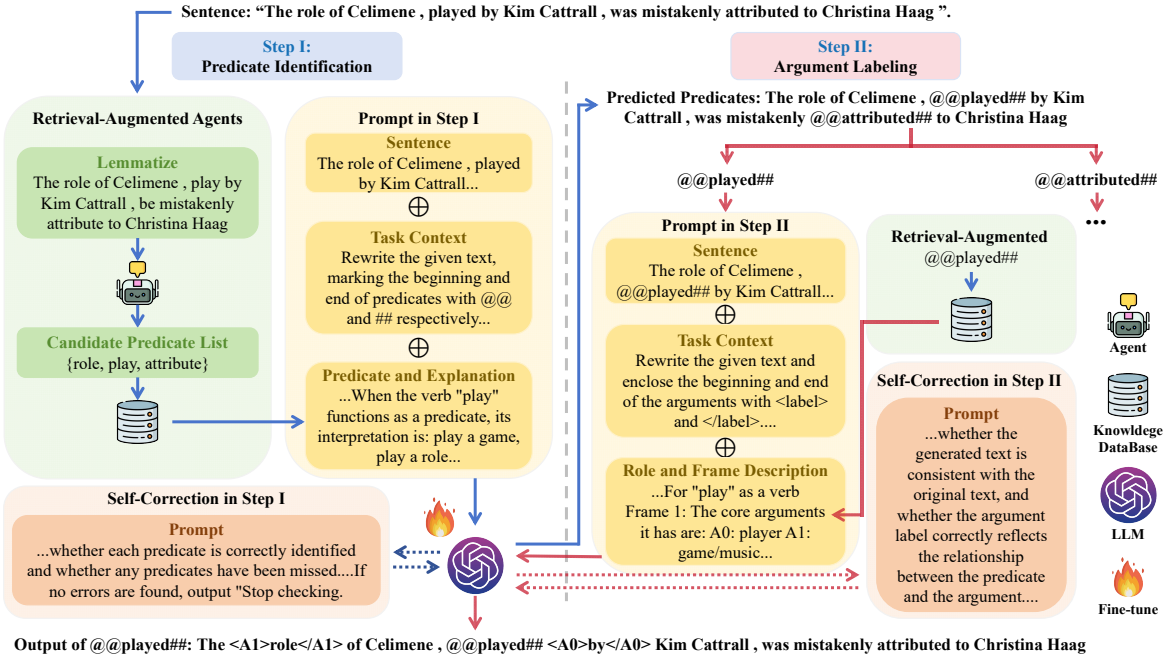


Figure 2: The two-step retrieval-augmented framework for SRL with self-correction mechanism. Step I performs predicate identification with retrieval-augmented prompting, while Step II conducts argument labeling through role and frame descriptions. Both steps incorporate self-correction modules to verify and refine the predictions.

Special tagging. To identify all arguments related to the given predicate and their corresponding roles, we insert special tags around each argument, enclosing the beginning and end with `<label>` and `</label>`. This produces sentences annotated with role tags for all gold-standard arguments associated with the predicate p_k:

$$Y^{p_k,a,r} = \{w_1, \ldots, <r_i>a_i</r_i>, \ldots, \ldots, @@p_k\#\#, \ldots, w_n\} \quad ,$$

where for $\forall(p_k, a, r) \subseteq (P, A, R)$, <r>a</r> is an element of $Y^{p_k,a,r}$.

Argument-role generation. Arguments are divided into: (1) **Core arguments** $\mathcal{R}^{\text{core}}$: essential for the predicate, typically labeled as ARGN (N = 0 ∼ 5), though these labels can be ambiguous without additional context. (2) **Adjunct arguments** $\mathcal{R}^{\text{adjunct}}$: Non-essential but supplementary, with consistent labels across predicates. To refine core argument labels, we use predicate-specific frame descriptions, reducing ambiguity and simplifying the label set. The complete argument label set $\mathcal{R}_k$ for the specific predicate p_k is:

$$\mathcal{R}_k = \mathcal{R}_k^{\text{core}} \cup \mathcal{R}^{\text{adjunct}},$$

where $\mathcal{R}_k^{\text{core}} \subseteq \mathcal{R}^{\text{core}}$ represents the labels derived from frame descriptions. This ensures $|\mathcal{R}_k| \leq |\mathcal{R}|$, and $\mathcal{R}$ denotes the overall label set.

To utilize this information, we retrieve all frame descriptions f_{desc} associated with the base form of the given predicate and include them in the prompt. This process can be expressed as:

$$\mathcal{D}_k^2 = \mathcal{D}^1 + \widetilde{Y}^p + C^a + \mathcal{R}_k + f_{\text{desc}},$$

where, C^a represents the argument labeling task context. The prompt $\mathcal{D}_k^2$ is then fed to the LLM to generate the argument-role result $\widetilde{Y}^{p_k,a,r}$ for the specific predicate p_k.

Training. The cross-entropy objective is used for parameter optimization during training:

$$\mathcal{L}_{\text{arg}} = -\sum_{k=1}^{z} \sum_{t=1}^{|Y^{p_k,a,r}|} \log P(Y_t^{p_k,a,r} | Y_{<t}^{p_k,a,r}, \mathcal{D}_k^2),$$

where z represents the number of gold predicates in $\widetilde{Y}^p$.

3.4 Self-Correction Mechanism

To address challenges such as inconsistency and illusion in LLM-generated outputs, we integrate a self-correction mechanism into our framework. This mechanism enables the LLM to autonomously identify, evaluate, and refine errors in its predictions through an iterative process.

Once the initial results for either predicate identification or argument labeling are generated, the

LLM is prompted to evaluate its outputs for inconsistencies or errors. If any issues are detected, the LLM adjusts its predictions accordingly. This process continues iteratively until either the maximum number of iterations N is reached or the model determines that no further corrections are necessary.

The self-correction mechanism for predicate identification is represented as follows:

$$\mathcal{D}_i^1 = \begin{cases} \mathcal{D}^1 + \widetilde{Y}^p + C_{iter}^p, & \text{if } i = 1 \\ \mathcal{D}_{i-1}^1 + \widetilde{Y}_{i-1}^p + C_{iter}^p + \widetilde{e}_{i-1}^p, & \text{else} \end{cases}$$

$$\widetilde{e}_i^p, \widetilde{Y}_i^p = \text{LLM}(\mathcal{D}_i^1),$$

where C_{iter}^p represents the context for self-correction in predicate identification, and $\widetilde{e}_i^p$ denotes the identified errors during the i-th iteration, which are used to refine the predictions.

During training, the gold-standard error e_i^p for the i-th iteration can be directly derived. Therefore, the gold-standard output for the i-th iteration is defined as: $Y_i^p = e_i^p + Y^p$. The self-correction loss for predicate identification is then computed as:

$$\mathcal{L}_{\text{sc}}^{\text{pred}} = -\sum_{i=1}^{N}\sum_{t=1}^{|Y_i^p|} \log P(Y_{i,t}^p | Y_{i,<t}^p, \mathcal{D}_i^1).$$

In the same way, we can also get the self-correction loss for argument labeling task $\mathcal{L}_{\text{sc}}^{\text{arg}}$. Finally, the overall self-correction loss is $\mathcal{L}_{\text{sc}} = \mathcal{L}_{\text{sc}}^{\text{pred}} + \mathcal{L}_{\text{sc}}^{\text{arg}}$.

3.5 Training Loss

The overall training loss $\mathcal{L}$ for the LLM is defined as a combination of three core components: the predicate identification loss $\mathcal{L}_{\text{pred}}$, the argument labeling loss $\mathcal{L}_{\text{arg}}$, and the self-correction loss $\mathcal{L}_{\text{sc}}$. This unified objective function is represented as:

$$\mathcal{L} = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{arg}} + \mathcal{L}_{\text{sc}}. \quad (1)$$

By jointly optimizing these three components, the framework achieves a balanced and robust learning process, enhancing the overall performance of the model in SRL tasks.

4 Experiments

4.1 Settings

Dataset. We conduct our experiments on three widely used datasets: CPB1.0 (Xue, 2008) for Chinese, CoNLL09 (Hajič et al., 2009) for both English and Chinese, and CoNLL12 (Pradhan et al., 2012) for English. The dataset statistics are summarized in Table 1. Among them, CPB1.0 and CoNLL12 are span-based SRL datasets, while CoNLL09 is dependency-based. For CoNLL12, following He et al. (2018), we extract data from OntoNotes (Pradhan et al., 2013) and adopt the standard data splits provided by the CoNLL12 shared task (Pradhan et al., 2012). Given the differences in predicate annotation across datasets, we process them accordingly. For CPB1.0 (Zh) and CoNLL09 (Zh), where frame files lack explicit predicate explanations, we use the frames as contextual input and employ GPT-4o-mini to generate predicate explanations within the given framework. To accelerate the selection of the optimal training step checkpoint, we randomly sample 800 sentences from the validation sets of CoNLL09 and CoNLL12 to create a new validation set. As the validation set of CPB1.0 (Zh) contains fewer than 800 sentences, resampling is not required.

Dataset		#Sent	#PA-Triples	#Roles
CPB1.0 (Zh)	Train	8,665	65,809	20
	Devel	549	4,445	16
	Test	983	8,001	17
CoNLL09 (En)	Train	38,770	36,5708	52
	Devel	1,334	12,918	32
	Re-Devel	800	7,690	29
	Test	2,396	21,634	36
	OOD	421	2,766	27
CoNLL09 (Zh)	Train	22,276	231,849	35
	Devel	1,762	18,554	24
	Re-Devel	800	8,302	24
	Test	2,556	27,712	26
CoNLL12 (En)	Train	75,187	566,718	62
	Devel	9,603	70,871	46
	Re-Devel	800	5,796	32
	Test	9479	72879	49

Table 1: Overview of datasets, where #PA-Triples denotes the number of automatic predicate-argument tuples: <predicate, argument, role>. Meanwhile, #Sent refers to the number of sentences, and #Roles indicates the number of roles.

Evaluation metrics. Following previous works on SRL (Zhou et al., 2020; Zhang et al., 2022), we exclude predicate sense disambiguation from our evaluation. The performance is measured based on atomic predicate-argument structures, represented as tuples in the form of <predicate, argument, role>. A tuple is deemed correct only if the predicate, argument span boundaries, and role all exactly match the gold-standard annotations. For span-based SRL datasets, we report precision, recall, and F_1 scores using the official evaluation script.[1] For dependency-based SRL datasets, we adopt the official CoNLL-2009 scoring script.[2]

[1] https://www.cs.upc.edu/ srlconll/st05/st05.html
[2] https://ufal.mff.cuni.cz/conll2009-st/scorer.html

Method	Without pre-identified predicates									With pre-identified predicates								
	CoNLL09						CoNLL12			CoNLL09						CoNLL12		
	WSJ			Brown						WSJ			Brown					
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Traditional Method																		
Li et al. (2020)	86.16	85.56	85.86	74.65	73.17	73.90	-	-	-	91.60	88.95	90.26	82.6	78.75	80.63	-	-	-
+ BERT	88.77	88.62	88.70	80.01	79.80	79.90	-	-	-	92.59	90.98	91.77	86.49	83.80	85.13	-	-	-
Zhou et al. (2020)	84.24	87.55	85.86	76.46	78.52	77.47	-	-	-	85.93	85.76	85.84	76.92	74.55	75.72	-	-	-
+ BERT	86.77	88.49	87.62	79.06	81.67	80.34	-	-	-	89.04	88.79	88.91	81.89	80.98	81.43	-	-	-
Zhang et al. (2022)	-	-	-	-	-	-	79.27	83.24	81.21	-	-	-	-	-	-	83.02	84.31	83.66
+ BERT	-	-	-	-	-	-	84.53	86.41	85.45	-	-	-	-	-	-	87.52	87.79	87.66
Fei et al. (2021a)	-	-	-	-	-	-	-	-	-	92.24	92.53	92.45	-	-	-	86.51	85.92	86.20
+ +RoBERTa	-	-	-	-	-	-	-	-	-	**92.89**	**92.80**	**92.83**	-	-	-	88.09	**88.83**	**88.59**
Fernández-González (2023)	85.90	88.00	86.90	74.40	76.40	75.40	-	-	-	90.20	90.50	90.40	80.60	80.50	80.60	-	-	-
+BERT	87.20	**89.80**	88.50	79.90	**81.80**	80.40	-	-	-	91.30	**91.60**	91.40	84.50	84.20	84.40	-	-	-
LLM-based Method																		
Sun et al. (2023)																		
ChatGPT+SimCSE kNN	-	-	-	-	-	-	-	-	-	-	-	84.80	-	-	-	-	-	82.80
Cheng et al. (2024)																		
Llama2-7B+3-shot	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1.83	10.67	3.13
Ours																		
Llama3-8B (Frozen)	2.63	1.85	2.17	3.14	2.89	3.01	6.29	7.59	6.88	4.19	4.71	4.43	4.70	6.87	5.58	21.91	16.64	18.91
Llama3-8B (Fine-tune)	**89.92**	88.23	**89.07**	**83.36**	79.04	**81.14**	**85.64**	85.59	**85.61**	92.78	91.02	91.89	**88.41**	**84.21**	**86.05**	**88.19**	88.06	88.12

Table 2: Main results on the CoNLL09 (En) in-domain (WSJ), out-of-domain (Brown), and CoNLL12 (En) test sets, where Ours experiments have the same settings except that they freeze or fine- tuned the LLM.

Model Details. The knowledge databases of each datasets are constructed using frame files provided by the respective datasets. Specifically, the frame file for CoNLL12 is derived from PropBank[3], which defines the frames of the predicates and provides explanations for their corresponding core arguments. The retrieval-augmented agent is designed based on rule matching, enabling it to retrieve explanations relevant to the SRL task by leveraging word-based queries. For word lemmatization, the agent employs the Lemminflect tool[4]. The agent's performance on predicate identification is reported in Appendix B. We utilize different LLMs for English and Chinese datasets. For English datasets, we adopt Llama-3-8B-Instruct[5] (Dubey et al., 2024), while for Chinese datasets, we use Qwen2.5-7B-Instruct[6] (Yang et al., 2024). We also evaluated different sizes of Qwen2.5 on CPB1.0, and the detailed results are provided in Appendix D. By default, we leverage the LLama-Factory framework[7] (Zheng et al., 2024) for parameter updates and fine-tune the LLMs using LoRA (Hu et al., 2022) to enable efficient learning, which is kept at default settings. The proportion of trainable parameters is provided in the Appendix E. Examples of prompts are provided in Appendix G.

Hyperparameters. All experiments are conducted on a single NVIDIA A800 Tensor Core GPU (80GB). Each setting is run three times with different random seeds, and the median evaluation scores are reported. We fine-tuned the LLMs for up to 200,000 steps with a learning rate of 1e-4, saving checkpoints every 20,000 steps. Best parameters retrieved from development sets are applied to each experiment on test sets. During training, the number of self-correction iterations N is fixed to 3 in both two steps.

Method	Without pre-identified predicates					
	CPB1.0			CoNLL09		
	P	R	F_1	P	R	F_1
Traditional Method						
Xia et al. (2019)	-	-	81.73	-	-	-
+ BERT	-	-	85.57	-	-	-
Li et al. (2020)	-	-	-	83.51	79.59	81.50
+ BERT	-	-	-	85.90	84.90	85.39
LLM-based Method						
Ours						
Qwen2.5-7B (Frozen)	3.22	3.59	3.39	2.16	1.67	1.88
Qwen2.5-7B (Fine-tune)	**89.24**	**87.40**	**88.31**	**87.56**	**86.02**	**86.78**

Method	With pre-identified predicates					
	CPB1.0			CoNLL09		
	P	R	F_1	P	R	F_1
Traditional Method						
Xia et al. (2019)	84.49	83.34	83.91	84.6	85.7	85.1
+ BERT	-	-	-	89.07	87.71	88.38
Li et al. (2020)	-	-	-	88.35	83.82	86.02
+ BERT	-	-	-	89.07	87.71	88.38
LLM-based Method						
Ours						
Qwen2.5-7B (Frozen)	9.45	8.46	8.93	4.51	3.42	3.89
Qwen2.5-7B (Fine-tune)	**90.60**	**88.18**	**89.37**	**89.56**	**87.82**	**88.68**

Table 3: Main results on CPB1.0 (Zh) and CoNLL09 (Zh) test sets.

Baselines. In this work, we select several recent works as baselines:

- **Traditional methods** are broadly categorized into transition-based (Fernández-González,

[3]https://github.com/propbank/propbank-frames
[4]https://github.com/bjascob/LemmInflect
[5]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[6]https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
[7]https://github.com/hiyouga/LLaMA-Factory

2023) and graph-based methods (Zhou et al., 2020; Fei et al., 2021a; Zhang et al., 2022), both of which have shown outstanding performance. Transition-based methods incrementally build SRL structures through a sequence of transition operations, while graph-based methods treat SRL as a graph parsing problem, explicitly modeling predicate-argument relationships.

- Sun et al. (2023) propose a few-shot SRL method using ChatGPT, assuming predicates are given. Sentence embeddings are used to retrieve examples via kNN, and arguments and their corresponding roles are identified by querying ChatGPT for the given predicate.

- Cheng et al. (2024) design a four-stage SRL pipeline for a given predicate: predicate disambiguation, role retrieval, argument labeling, and post-processing. They evaluate the effectiveness of LLMs in a few-shot setting.

4.2 Main Results

The main results are presented in Table 2 and Table 3. Our study focuses on generating both predicates and their associated arguments, while prior works (Sun et al., 2023; Cheng et al., 2024) typically assume predicates are pre-identified and focus solely on argument recognition. To ensure a fair comparison, we also report results under the predicate-given setting.

The results reveal that using LLMs for SRL tasks is inherently challenging. When the LLM is frozen, its F_1 scores range from 2 to 25, significantly lower than traditional methods. This underscores the difficulty of directly applying LLMs to SRL, particularly in settings without pre-identified predicates.

However, despite these challenges, our retrieval-augmented LLM-based framework consistently outperforms traditional methods in many settings. For example, in the setting without pre-identified predicates, our method achieves improvements over the previous state-of-the-art by $+\mathbf{0.37}$ F_1 on CoNLL09-WSJ (En), $+\mathbf{0.74}$ F_1 on CoNLL09-Brown (En), and $+\mathbf{0.16}$ F_1 on CoNLL12 (En). The larger gain on the out-of-domain CoNLL09-Brown (En) test set highlights the strong domain adaptation capabilities of our approach. Similarly, on Chinese datasets, our method achieves $+\mathbf{2.74}$ F_1 on CPB1.0 (Zh) and $+\mathbf{1.39}$ F_1 on CoNLL09 (Zh) without pre-identified predicates.

Method	CPB1.0	CoNLL09
Ours	**88.31**	**86.78**
-w/o retrieval-augmented agent	85.92	82.86
-w/o role and frame description	86.43	84.02
-w/o self-correction	87.05	85.47
-w/o all	80.39	77.34

Table 4: The performance of each component based on the CPB1.0 (Zh) and CoNLL09-WSJ (En) test sets.

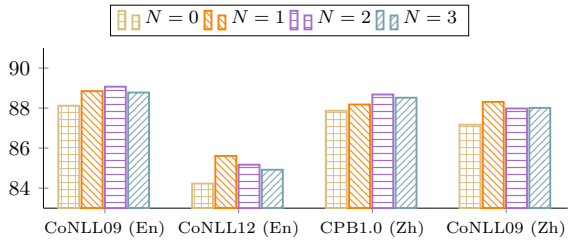
Fei et al. (2021a) incorporate additional syntactic information and achieve superior results on CoNLL09-WSJ (En) and CoNLL12 (En) under the setting where predicates are given. However, without leveraging syntactic information, their reported F1 scores are 92.05 on CoNLL09-WSJ (En) and 85.79 on CoNLL12 (En). In comparison, our approach attains a comparable F1 score on CoNLL09-WSJ (En) and achieves $+2.33$ F_1 on CoNLL12 (En). These results demonstrate that our framework effectively addresses the challenges faced by SRL and achieves robust performance in different languages and domains.

When compared with other LLM-based approaches, our method exhibits clear advantages. Prior works (Sun et al., 2023; Cheng et al., 2024) conducted experiments under the predicate-given setting, which simplifies the SRL task. Even in this setting, our method achieves substantial improvements. For instance, Sun et al. (2023) report F_1 scores of 84.8 and 82.8 on CoNLL09-WSJ (En) and CoNLL12 (En), respectively, while our framework achieves $+\mathbf{7.09}$ F_1 and $+\mathbf{5.32}$ F_1 improvements on these datasets. These results strongly validate the effectiveness of our two-step retrieval-augmented framework and demonstrate its ability to outperform existing LLM-based methods.

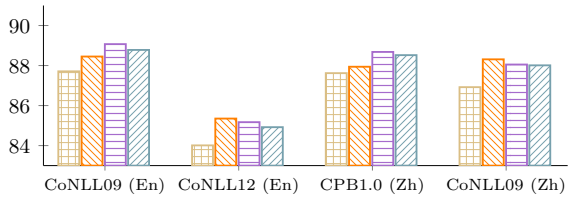
Overall, the results highlight that while SRL remains a challenging task for LLMs, our proposed retrieval-augmented framework not only addresses these challenges but also surpasses both traditional methods and other LLM-based approaches.

4.3 Analysis

Ablation experiments. To assess the contribution of each proposed component, we conducted ablation experiments by removing them individually and evaluating performance on the CPB1.0 (Zh) and CoNLL09-WSJ (En) test sets, as shown in Table 4. The results confirm that each component significantly enhances performance, underscoring their importance in achieving optimal results.



(a) F_1 scores for predicate recognition with self-correction across different iteration number N



(b) F_1 scores for the whole SRL task with self-correction across different iteration number N

Figure 3: The results of varying iterations of self-correction on predicate identification and SRL. In (a), the number of iterations for predicate identification varies, while the iterations for argument labeling are fixed at 3. In (b), the number of iterations for the two steps is kept consistent.

Num		Outputs
1	**Before**	That country recently @@ 200,000 tons of sugar.
	After	That country recently @@bought## 200,000 tons of sugar.
2	**Before**	That's the biggest @@risk## of all.
	After	That's the <EXT>biggest</EXT> @@risk## of all.
3	**Before**	That <A0>that country</A0> <AD>recently</ADV> @@bought## <A1>200,000 tons of sugar</A1>.
	After	<A0>That country</A0> <ADV>recently</ADV> @@bought## <A1>200,000 tons of sugar</A1>.
4	**Before**	<A0>We </A0> @@need## to have PNC>to</PNC> spend $ 5 billion properly.
	After	<A0>We </A0> @@need## to have <PNC>to</PNC> spend $ 5 billion properly.
5	**Before**	<A0>You</A0> are not @@thinking## of <A1>going</A1>, right?
	After	<A0>You</A0> are not @@thinking## <A1> of going</A1>, right?

Table 5: Cases about self-correction impact in English datasets, where **Before** denotes the output before self-correction and **After** denotes that after self-correction. Cases about the impact of self-correction on Chinese datasets can be found in the Appendix C.

Among these, the retrieval-enhanced agent and the role and frame description stand out as key contributors. The retrieval-enhanced agent leverages external resources to generate a list of candidate predicates along with their explanations, thereby improving predicate recognition accuracy. This improvement cascades into more accurate argument labeling. Similarly, the role and frame description aids the LLM in better understanding predicate meanings and role semantics, further boosting overall performance. In contrast, self-correction had relatively little impact. As the model adapts to the task with increased training steps, self-correction becomes less critical, especially for simpler sentence annotations. Removing these components results in performance declines of $\mathbf{7.92}\%$ and $\mathbf{9.44}\%$, respectively, demonstrating the overall effectiveness of our proposed method.

The impact of iteration N. The impact of iteration number N on performance is illustrated in Figure 3, which presents F_1 scores for both predicate identification and the whole SRL task under varying iteration numbers.

The results show that increasing the number of iterations initially improves performance but leads to declines when N becomes too large. This trend could be attributed to two factors: (1) the varying difficulty levels across datasets, which require different optimal iteration counts for best performance, and (2) the potential inaccuracies in error feedback from the LLM, where excessive iterations may amplify deviations from the desired output instead of correcting earlier errors. Notably, for span-based datasets such as CPB1.0 (Zh) and CoNLL12 (En), the optimal performance is achieved at $N = 1$, while for the dependency-based dataset CoNLL09, the best results are observed at $N = 2$. Furthermore, the performance trends for the entire SRL task closely mirror those for predicate identification, underscoring the critical role of accurate predicate identification in enhancing the subsequent argument labeling process.

A case study of self-correction. Table 5 illustrates the impact of self-correction on predicate identification and argument labeling. The cases show that self-correction effectively mitigates errors caused by formatting inconsistencies, illusions, and the inherent complexity of the SRL task.

First, self-correction addresses formatting errors and inconsistencies between the generated text and the original. For instance, in Case 1, the predicate "`bought`" was missing. After self-correction, the LLM successfully restores the missing predicate and aligns the output with the in-

tended format. Second, self-correction resolves errors in role tags. As shown in Case 3 and Case 4, the LLM initially generates inconsistent role tags <AD>recently</ADV> and incomplete role tags PNC>to</PNC>, which are corrected through self-correction, resulting in a more coherent and accurate output. Finally, the iterative nature of self-correction helps refine complex SRL outputs. For example, in Case 5, the argument was initially labeled as "of <A1>going</A1>", where the boundary of the argument tag was misplaced. After self-correction, the LLM correctly identifies and adjusts the argument to "<A1>of going</A1>", ensuring an accurate representation of the argument's scope.

5 Conclusion

In this work, we proposed a novel retrieval-augmented framework aimed at bridging the performance gap between large language models (LLMs) and traditional encoder-decoder models in semantic role labeling (SRL). Through the integration of retrieval-augmented agents, which leverage external knowledge about predicate and argument structures, and a self-correction mechanism for iterative refinement, our method addressed key challenges in SRL, including the complexity of predicate-argument structures, the need for domain-specific linguistic knowledge, and the illusion issues often associated with LLM outputs. Experiments conducted on three widely-used benchmarks—CPB1.0, CoNLL-2009, and CoNLL-2012—in both English and Chinese demonstrated that our approach achieved state-of-the-art results. Notably, it marked the first instance of an LLM-based method outperforming traditional approaches across complete SRL tasks. These results highlight the transformative potential of combining LLMs with structured reasoning and external knowledge for tackling complex linguistic tasks, paving the way for further advancements in semantic understanding.

Limitations

While our proposed framework achieved state-of-the-art performance in SRL, there are several limitations that warrant further exploration. First, the current self-correction strategy is relatively simple, relying on iterative refinement without explicitly modeling the reasoning process. This simplicity stems from the need to maintain computational efficiency during large-scale training. Future work could incorporate advanced techniques, such as chain-of-thought prompting, to enable more structured and interpretable self-correction. Second, the retrieval-augmented agent currently employs a rule-based traversal approach for candidate predicate retrieval. This design ensures that the candidate predicates are as comprehensive as possible, providing the LLM with sufficient context for accurate predicate identification. However, this approach may limit flexibility and scalability. In the future, we plan to explore the use of generative large language models to dynamically generate candidate predicates, potentially improving both efficiency and accuracy. Third, the loss function in Equation 1 assigns equal weights to all components, which may not fully capture the varying importance of predicate identification, argument labeling, and self-correction. Further studies could explore the impact of different weighting strategies on overall performance. Finally, our approach primarily targets sentence-level SRL, and its scalability to document-level or cross-lingual SRL remains an open challenge, which we aim to address in future research.

Ethical Statement

This work does not involve the use of sensitive or private data, and all experiments were conducted on publicly available datasets (CPB1.0, CoNLL-2009, and CoNLL-2012) following their respective usage guidelines. We have obtained the license for these datasets. The proposed framework is designed to advance SRL research and does not inherently pose risks of misuse. However, as with any language model-based system, there is a potential for generating biased or incorrect outputs due to limitations in the training data or model design. We encourage responsible use of this technology and advocate for ongoing efforts to mitigate bias and ensure fairness in NLP applications. Additionally, no human subjects were involved in this research, and no ethical concerns were identified during the course of this study.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. This work is supported by the National Natural Science Foundation of China (Grant Nos. 62176180), the Shenzhen Science and Technology Program (Grant Nos. ZDSYS20230626091203008

and JCYJ20241202123503005), and the Shenzhen College Stability Support Plan (Grant Nos. GXWD20231130140414001).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv*.

Thorsten Barnickel, Jason Weston, Ronan Collobert, Hans-Werner Mewes, and Volker Stümpflen. 2009. Large scale application of neural network based semantic role labeling for automated relation extraction from biomedical texts. *PloS one*, 4(7):e6393.

Emanuele Bastianelli, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. 2013. Textual inference and meaning representation in human robot interaction. In *Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora*, pages 65–69.

Emanuele Bastianelli, Giuseppe Castellucci, Danilo Croce, Roberto Basili, and Daniele Nardi. 2014. Effective and robust natural language understanding for human-robot interaction. In *Proceedings of the Twenty-First European Conference on Artificial Intelligence*, page 57–62.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. 2021. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3786–3793.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Huiyao Chen, Xinxin Li, Meishan Zhang, and Min Zhang. 2024. Semantic role labeling from Chinese speech via end-to-end learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8898–8911.

Huiyao Chen, Meishan Zhang, Jing Li, Min Zhang, Lilja Øvrelid, Jan Hajič, and Hao Fei. 2025. Semantic role labeling: A systematical survey. *arXiv preprint arXiv:2502.08660*.

Ning Cheng, Zhaohui Yan, Ziming Wang, Zhijie Li, Jiaming Yu, Zilong Zheng, Kewei Tu, Jinan Xu, and Wenjuan Han. 2024. Potential and limitations of llms in capturing structured semantics: A case study on SRL. In *Advanced Intelligent Computing Technology and Applications - 20th International Conference, ICIC 2024, Tianjin, China, August 5-8, 2024, Proceedings, Part I (LNAI)*, volume 14875, pages 50–61.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2010. Semantic role labeling for open information extraction. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, pages 52–60.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the sixth international conference on Knowledge capture*, pages 113–120.

Angel Daza and Anette Frank. 2018. A sequence-to-sequence model for semantic role labeling. In *Proceedings of the Third Workshop on Representation Learning for NLP*, pages 207–216.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*.

Richard Evans and Constantin Orasan. 2019. Sentence simplification for semantic role labelling and information extraction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 285–294.

Hao Fei, Shengqiong Wu, Yafeng Ren, Fei Li, and Donghong Ji. 2021a. Better combine them together! integrating syntactic constituency and dependency representations for semantic role labeling. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 549–559, Online. Association for Computational Linguistics.

Hao Fei, Meishan Zhang, Bobo Li, and Donghong Ji. 2021b. End-to-end semantic role labeling with neural transition-based model. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 12803–12811. AAAI Press.

Daniel Fernández-González. 2023. Transition-based semantic role labeling with pointer networks. *Knowledge-Based Systems*, 260:110127.

Daniel Fernández-González and Carlos Gómez-Rodríguez. 2020. Transition-based semantic dependency parsing with pointer networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7035–7046, Online. Association for Computational Linguistics.

Sourav Garg, Niko Sünderhauf, Feras Dayoub, Douglas Morrison, Akansel Cosgun, Gustavo Carneiro, Qi Wu, Tat-Jun Chin, Ian D. Reid, Stephen Gould, Peter Corke, and Michael Milford. 2020. Semantics for robotic mapping, perception and interaction: A survey. *Foundations and Trends in Robotics*, 8(1-2):1–224.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 512–520.

Hongchao Gu, Dexun Li, Kuicai Dong, Hao Zhang, Hang Lv, Hao Wang, Defu Lian, Yong Liu, and Enhong Chen. 2025. Rapid: Efficient retrieval-augmented long text generation with writing planning and information discovery. *arXiv preprint*.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 364–369.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*.

Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Xing Wang, Shuming Shi, and Zhaopeng Tu. 2023. Is chatgpt a good translator? yes with gpt-4 as the engine. *arXiv*.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907.

Tao Lei, Yuan Zhang, Lluís Màrquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1150–1160.

Zuchao Li, Shexia He, Jiaxun Cai, Zhuosheng Zhang, Hai Zhao, Gongshen Liu, Linlin Li, and Luo Si. 2018. A unified syntax-aware framework for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2401–2411, Brussels, Belgium. Association for Computational Linguistics.

Zuchao Li, Hai Zhao, Rui Wang, and Kevin Parnow. 2020. High-order semantic role labeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1134–1151.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 716–724.

Wenpeng Lu, Rui Yu, Shoujin Wang, Can Wang, Ping Jian, and Heyan Huang. 2021. Sentence semantic matching based on 3d cnn for human–robot language interaction. *ACM Transactions on Internet Technology.*, 21(4).

Chunchuan Lyu, Shay B. Cohen, and Ivan Titov. 2019. Semantic role labeling with iterative structure refinement. 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 1071–1082.

Diego Marcheggiani, Jasmijn Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 486–492.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Daniel Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 581–588.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 12–21.

Chen Shi, Shujie Liu, Shuo Ren, Shi Feng, Mu Li, Ming Zhou, Xu Sun, and Houfeng Wang. 2016. Knowledge-based semantic embedding for machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2245–2254.

Xiaofei Sun, Linfeng Dong, Xiaoya Li, Zhen Wan, Shuhe Wang, Tianwei Zhang, Jiwei Li, Fei Cheng, Lingjuan Lyu, Fei Wu, and Guoyin Wang. 2023. Pushing the limits of chatgpt on NLP tasks. *CoRR*.

Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond Mooney. 2020. Jointly improving parsing and perception for natural language commands through human-robot dialog. *Journal of Artificial Intelligence Research*, 67:327–374.

Cynthia A. Thompson, Siddharth Patwardhan, and Carolin Arnold. 2004. Generative models for semantic role labeling. In *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 235–238.

WenHao Wang, Xiaoyu Liang, Rui Ye, Jingyi Chai, Siheng Chen, and Yanfeng Wang. 2024. KnowledgeSG: Privacy-preserving synthetic text generation with knowledge distillation from server. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7677–7695.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Qingrong Xia, Zhenghua Li, and Min Zhang. 2019. A syntax-aware multi-task learning framework for Chinese semantic role labeling. 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 5382–5392.

Kun Xu, Haochen Tan, Linfeng Song, Han Wu, Haisong Zhang, Linqi Song, and Dong Yu. 2020. Semantic role labeling guided multi-turn dialogue rewriter. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6632–6639.

Nianwen Xue. 2008. Labeling chinese predicates with semantic roles. *Computational Linguistics*, 34(2):225–255.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *CoRR*.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 201–206.

Deniz Yuret, Mehmet Ali Yatbaz, and Ahmet Engin Ural. 2008. Discriminative vs. generative approaches in semantic role labeling. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 223–227.

Yu Zhang, Qingrong Xia, Shilin Zhou, Yong Jiang, Guohong Fu, and Min Zhang. 2022. Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4212–4227.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020.

Semantics-aware BERT for language understanding. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 9628–9635.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033.

Junru Zhou, Zuchao Li, and Hai Zhao. 2020. Parsing all: Syntax and semantics, dependencies and spans. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4438–4449.

A Comparison of Unified and Separate Training Strategies

Training Strategy	Method	Predicate F_1 (%)	SRL F_1 (%)
Separate Predicate	Traditional	94.77	-
Separate Argument	Traditional	100.00	85.80
Unified	Traditional	93.80	86.14
Separate Predicate	Ours	97.33	-
Separate Argument	Ours	100.00	88.55
Unified	Ours	97.15	88.04

Table 6: Performance comparison of separate vs. unified training strategies across traditional and LLM-based approaches. "Separate" denotes independent training of predicate identification and argument labeling, while "Unified" refers to joint end-to-end optimization.

To validate our unified LLM-based training approach, we compare it against separate training strategies for predicate identification and argument labeling. As shown in Table 6, while separate training achieves marginally higher SRL F_1 scores (88.55% vs. 88.04%, +0.51% improvement), the unified approach offers several key advantages:

(1) **Streamlined pipeline**: Single-stage optimization eliminates the complexity of coordinating separate components. (2)**Consistent performance**: Our unified LLM-based method (88.04%) substantially outperforms traditional unified approaches (86.14%). (3) **Practical deployment**: Unified training simplifies model deployment and maintenance. Notably, our LLM-based approach demonstrates superior performance over traditional methods in both strategies, with particularly strong gains in predicate identification (97.33% vs. 94.77% for separate, 97.15% vs. 93.80% for unified).

These results justify our design choice of unified prompt-based formulation, achieving competitive performance while maintaining practical advantages for real-world deployment.

B Predicate Identification Performance of Rule-based Agent

Dataset	Split	#Predicates	#Missed	Hit Rate (%)
CPB1.0	Train	30,220	1	100.00
	Test	3,513	0	100.00
CoNLL09 (En)	Train	177,971	33	99.98
	WSJ Test	10,498	73	99.30
	Brown Test	1,259	63	95.00
CoNLL09 (Zh)	Train	102,803	124	99.88
	Test	12,282	20	99.84
CoNLL12	Train	75,187	0	100.00
	Test	9,479	0	100.00

Table 7: Predicate identification performance of the rule-based retrieval agent across different datasets. #Predicates and #Missed indicate the total number of predicates and failed retrievals, respectively.

Our framework employs a rule-based retrieval agent for predicate identification, which operates through lemmatization and exact string matching against a predefined framework database. While conceptually simple, this component plays a crucial role in supporting subsequent SRL processing.

Table 7 presents the predicate identification performance across multiple standard datasets. A predicate is considered successfully identified when the agent correctly matches both the position and form of the gold-standard annotation.

The results demonstrate excellent performance across most datasets, with hit rates exceeding 99% in 6 out of 8 test scenarios. Key observations include: (1) **Perfect accuracy**: CPB1.0 and CoNLL12 achieve 100% hit rates on both training and test sets, indicating strong coverage for these datasets. (2) **Consistent performance**: Most datasets maintain hit rates above 99.8%, demonstrating the reliability of the rule-based approach for in-domain scenarios. (3) **Domain sensitivity**: The Brown shows reduced performance (95.00%), indicating challenges with domain shift and lexical variability compared to the WSJ (99.30%).

The performance gap between Brown and WSJ test sets highlights the agent's sensitivity to out-of-domain data, suggesting potential benefits from incorporating LLM-based predicate retrieval for enhanced robustness across diverse domains. Despite these constraints, the rule-based agent provides reliable predicate identification for most scenarios while maintaining computational efficiency.

C Self-correction in Chinese datasets

Num		Outputs
1	**Before**	集团在上海追加 @@投资## 二千四百万美元。 The group in Shanghai added an @@investment## of 24 million dollars.
	After	集团在上海 @@追加## 投资二千四百万美元。 The group in Shanghai @@added## an investment of 24 million dollars.
2	**Before**	<A0>她</A0> @@说##，<A1>“化工重点项目建设进展较快。</A1> <A0>She</A0> @@said##, <A1>“The construction progress of key chemical engineering projects is relatively fast.</A1>
	After	<A0>她</A0> @@说##，<A1>“化工重点项目建设进展较快。”</A1> <A0>She</A0> @@said##, <A1>“The construction progress of key chemical engineering projects is relatively fast.”</A1>
3	**Before**	城建成为 <A0>外商</A0>@@投资## 青海新热点。 Urban construction has become a new hotspot for <A0> foreign investors </A0> to @@invest## Qinghai.
	After	城建成为 <A0>外商</A0>@@投资## <A1>青海</A1> 新热点。 Urban construction has become a new hotspot for <A0> foreign investors </A0> to @@invest## <A1>Qinghai</A1>.
4	**Before**	<A1>东亚经济<A1> <ADV>一定</ADV> 能够 <ADV>继续向前</ADV> @@发展##。 <A1>The East Asian economy</A1> <ADV>will certainly</ADV> able to <ADV>continue moving forward </ADV> @@develop##.
	After	<A1>东亚经济<A1> <ADV>一定</ADV> 能够继续 <ADV>向前</ADV> @@发展##。 <A1>The East Asian economy</A1> <ADV>will certainly</ADV> able to continue <ADV> moving forward </ADV> @@develop##.

Table 8: Cases about self-correction impact in Chinese datasets, where **Before** denotes the output before self-correction and **After** denotes that after self-correction. Each Chinese case is followed by its corresponding English translation.

Table 8 illustrates how self-correction improves predicate identification and argument labeling in Chinese datasets, addressing issues such as formatting errors, inconsistencies, and illusions, as described in Section 8. For example, in Case 1, the predicate "追加" (added) was initially mislabeled as "投资" (investment). After applying the self-correction mechanism, the model successfully rectified this error, aligning the output with the original text. Similarly, in Case 4, the model initially failed to include the right boundary of the argument labeled "ADV". Through self-correction, the model refined the argument's scope, ensuring a more accurate representation. These examples demonstrate that the self-correction mechanism in Chinese datasets operates similarly to its English counterparts by effectively enhancing the accuracy and coherence of the SRL task.

While the self-correction framework remains consistent across languages, its adaptability allows LLMs to leverage linguistic characteristics unique to each language. In English datasets (as shown in Table 5), self-correction often focuses on resolving ambiguities in argument roles and refining argument boundaries. For instance, English SRL cases frequently involve correcting role tags like "ADV" or handling complex discontinuous spans. In contrast, Chinese SRL tends to encounter challenges related to predicate disambiguation and ensuring the semantic alignment of predicates and arguments, as Chinese predicates often carry more implicit meanings. This flexibility highlights the ability of our framework to leverage the learning capabilities of the LLMs to identify and adapt the most critical self-correction directions for optimal performance based on the specific characteristics of each language.

D Impact of the LLM scale

Method	P	R	F_1
Ours			
- GPT-4o	8.28	12.79	10.05
- DeepSeek-V3-Chat	14.72	39.37	21.43
- Qwen2.5-72B (Frozen)	13.23	16.79	14.80
- Qwen2.5-1.5B (Fine-tune)	83.89	82.36	83.12
- Qwen2.5–3B (Fine-tune)	85.87	83.20	84.63
- Qwen2.5-7B (Fine-tune)	89.24	87.40	88.31
- Qwen2.5-14B (Fine-tune)	**89.72**	**87.55**	**88.62**

Table 9: Performance comparison of LLMs with varying parameter sizes on CPB1.0 dataset without predicate pre-identification. Results show Precision (P), Recall (R), and F_1 for both frozen and fine-tuned models.

Table 9 demonstrates the critical importance of fine-tuning and reveals interesting scaling patterns for SRL tasks. Our analysis yields several key findings:

Frozen models struggle regardless of scale. Even large-scale models perform poorly without fine-tuning: (1) GPT-4o achieves only 10.05% F_1 score, (2) Qwen2.5-72B (Frozen) reaches 14.80% F_1, likely benefiting from its Chinese optimization that aligns with the CPB1.0 dataset, and (3) DeepSeek-V3-Chat shows the best frozen performance at 21.43% F_1, suggesting more effective built-in strategies for SRL tasks.

Fine-tuning enables dramatic performance gains. The transition from frozen to fine-tuned models shows remarkable improvement, with Qwen2.5-1.5B jumping to 83.12% F_1 score. This 69-point improvement underscores that task-specific adaptation is essential for specialized tasks like SRL, regardless of the model's general capabilities.

Scaling benefits emerge with fine-tuning. Among fine-tuned models, larger sizes consistently improve performance: Qwen2.5-3B (84.63% F_1) $\rightarrow$ Qwen2.5-7B (88.31% F_1) $\rightarrow$ Qwen2.5-14B (88.62% F_1). However, the marginal gain diminishes significantly beyond 7B parameters (only 0.31% improvement from 7B to 14B), suggesting diminishing returns at larger scales.

Implications for model selection. The results indicate that (1) fine-tuning is non-negotiable for SRL tasks, as even 72B frozen models underperform 1.5B fine-tuned ones, (2) moderate scaling (1.5B $\rightarrow$ 7B) provides substantial benefits, but (3) further scaling beyond 7B offers limited gains, emphasizing the need for balanced strategies that combine appropriate model size with effective task-specific adaptations rather than relying solely on parameter scaling.

E Discussion on Trainable Parameters

Method	Model Type	#Trainable	Total
Li et al. (2020)	BERT-large	>355M (100%)	>355M
Zhou et al. (2020)	BERT-large	510.7M (100%)	510.7M
Zhang et al. (2022)	BERT-large	338.6M (100%)	338.6M
Fernández-González (2023)	BERT-large	>360M (100%)	>360M
Ours	Qwen2.5-7B	**20.2M (0.26%)**	7.6B
Ours	LLaMA3-8B	**21.0M (0.26%)**	8.1B

Table 10: Parameter efficiency comparison between BERT-based and LLM-based SRL models. #Trainable shows trainable parameters during fine-tuning, with percentages indicating the proportion of total model parameters requiring updates.

While LLM-based approaches utilize models with substantially larger total parameter counts, they achieve remarkable parameter efficiency during fine-tuning through parameter-efficient adaptation techniques such as LoRA.

Table 10 presents a comprehensive comparison between established BERT-based SRL methods and our LLM-based approach across key efficiency metrics. The results reveal a striking contrast in training requirements: (1) **BERT-based models require full fine-tuning**: Traditional approaches fine-tune 100% of their parameters, ranging from 338M to over 510M trainable parameters across different implementations. (2) **LLM-based models achieve extreme parameter efficiency**: Our method fine-tunes only 20-21M parameters (approximately 0.26% of total model size), representing a 15-25$\times$ reduction in trainable parameters compared to BERT-based approaches. (3) **Efficiency scales favorably**: Despite using 7-8B parameter base models, our approach requires fewer trainable parameters than any BERT-based method, demonstrating superior training efficiency.

We will explore more efficient methods in future work. For now, our primary goal is to establish a strong baseline and bridge the SRL task into the decoder-only LLM paradigm. This direction aligns with the trend of leveraging LLMs as a unified solution for various language understanding tasks.

F Description of Our SRL Algorithm

To further clarify the implementation details of our proposed framework, we provide a detailed explanation of the retrieval-augmented SRL with self-correction algorithm, as outlined in Algorithm 1. This algorithm systematically integrates retrieval-augmented agents and self-correction mechanisms into the SRL process, ensuring both accuracy and consistency in predicate-argument-role identification. Below, we describe the key stages of the algorithm:

Predicate identification with retrieval. The process begins by lemmatizing the input sentence to normalize its tokens. A retrieval-augmented agent then generates a list of candidate predicates by analyzing the lemmatized sentence and retrieving relevant contextual information from a knowledge database. For each candidate predicate, the agent retrieves corresponding explanations, which are combined with the input sentence and task-specific context to form a prompt. This prompt is fed into the LLM to identify predicates.

Predicate self-correction. After the initial predicate identification, a self-correction mechanism iteratively refines the predictions. At each iteration, the LLM evaluates its outputs, identifies potential errors, and updates its predictions accordingly. This process continues until no further errors are detected or the maximum number of iterations is reached.

Argument labeling. For each identified predicate, the algorithm retrieves role sets and frame

descriptions from the knowledge database. These are used as contextual prompts to guide the LLM in labeling arguments and assigning semantic roles.

Argument self-correction. Similar to predicate self-correction, the algorithm applies an iterative self-correction mechanism to refine argument labeling results. The LLM evaluates its outputs for consistency and correctness, making adjustments as needed.

G Prompts of Our SRL Framework

This section provides the specific prompts used in our approach to ensure reproducibility. Detailed prompts for the Chinese dataset will be included in our code repository.

In the prompt templates: (1) fixed prompts are displayed in black. (2) Input text is highlighted in deep red. (3) The candidate predicate list, along with explanations retrieved by the retrieval-augmented agent, is shown in purple. (4) The output generated by the LLM is presented in green.

These conventions are designed to make the prompts clear and easy to follow.

Algorithm 1: Retrieval-Augmented SRL with Self-Correction

Input: Input sentence: $X = w_1, w_2, ..., w_n$; Knowledge database: K_D; Maximum correction iterations: N; Task-specific contexts: C (C^p: Predicate identification context, C^p_{iter}: Predicate self-correction context, C^a: Argument labeling context, C^a_{iter}: Argument self-correction context)

Output: SRL triples $(P, A, R) = \{(p_1, a_1, r_1), ..., (p_m, a_m, r_m)\}$

/* Stage 1: Predicate Identification with Retrieval */

$X_{base} \leftarrow \text{Lemmatize}(w_1, w_2, ..., w_n)$;

Generate candidate predicates $\hat{P} = \{\hat{p}_1, ..., \hat{p}_k\}$ from X_{base} by the retrieval-argumented agent;

Retrieve explanations $E_{\hat{p}_i}$ for each $\hat{p}_i$ from K_D;

$\mathcal{D}^1 \leftarrow X + C^p + \{(\hat{p}_i, E_{\hat{p}_i}) | i = 1, 2, ..., k\}$;

Generate initial predicate results: $\widetilde{Y}^p \leftarrow \text{LLM}(\mathcal{D}^1)$;

/* Predicate Self-Correction */

for $i \leftarrow 1$ **to** N **do**

  if $i = 1$ **then**

    $\mathcal{D}^1_i \leftarrow \mathcal{D}^1 + \widetilde{Y}^p + C^p_{iter}$;

  else

    $\mathcal{D}^1_i \leftarrow \mathcal{D}^1_{i-1} + \widetilde{Y}^p_{i-1} + C^p_{iter} + \widetilde{e}^p_{i-1}$;

  end

  Generate errors and updated results: $\widetilde{e}^p_i, \widetilde{Y}^p_i \leftarrow \text{LLM}(\mathcal{D}^1_i)$;

  if *no errors detected* **then**

    break;

  end

end

/* Stage 2: Argument Labeling */

foreach *predicate* p_k *in final* $\widetilde{Y}^p$ **do**

  Get role set $\mathcal{R}_k = \mathcal{R}^{\text{core}}_k \cup \mathcal{R}^{\text{adjunct}}$;

  Retrieve frame descriptions f_{desc} for p_k from K_D;

  $\mathcal{D}^2_k \leftarrow \mathcal{D}^1 + \widetilde{Y}^{\text{p}} + C^a + \mathcal{R}_k + f_{\text{desc}}$;

  Generate initial argument results: $\widetilde{Y}^{\text{p}_k,a,r} \leftarrow \text{LLM}(\mathcal{D}^2_k)$;

  /* Self-Correction Arguments for Current Predicate */

  for $i \leftarrow 1$ **to** N **do**

    if $i = 1$ **then**

      $\mathcal{D}^2_{k,i} \leftarrow \mathcal{D}^2_k + \widetilde{Y}^{p_k,a,r} + C^a_{iter}$;

    else

      $\mathcal{D}^2_{k,i} \leftarrow \mathcal{D}^2_{k,i-1} + \widetilde{Y}^{p_k,a,r}_{i-1} + C^a_{iter} + \widetilde{e}^{p_k,a,r}_{i-1}$;

    end

    Generate errors and updated results: $\widetilde{e}^{p_k,a,r}_i, \widetilde{Y}^{p_k,a,r}_i \leftarrow \text{LLM}(\mathcal{D}^2_{k,i})$;

    if *no errors detected* **then**

      break;

    end

  end

end

return *SRL triples from all final* $\widetilde{Y}^{p_k,a,r}$

Prompt $\mathcal{D}^1$: Predicate Identification

System: You are a helpful assistant who has a background in linguistics and is good at understanding texts, especially skilled in semantic role labeling recognition.

User: Semantic Role Labeling (SRL) aims to identify predicates in a sentence and assign roles to their arguments. A predicate refers to the core word or phrase in a sentence that conveys an action, event, or state and serves as the focus for other elements in the sentence.
Text: What was the , purpose and goal of this campaign ?
For the predicate indentification task, what are the predicates in the given text?
Possible predicate results in the text are: "What @@was## the , @@purpose## and goal of this @@campaign## ?", where predicates are specified by @@ and ##.
When the verb "purpose" functions as a predicate, its interpretation is: purpose.
When the verb "be" functions as a predicate, its interpretation is: copula, existential, auxiliary, be like: multiword expression akin to "say".
When the verb "campaign" functions as a predicate, its interpretation is: work towards a goal.
Based on the given possible predicate results and interpretations, rewrite the given text, marking the beginning and end of predicates with @@ and ## respectively. Note that words not present in the predicate results may also be predicates.
LLM: Predicate identification result.

Prompt C^p_{iter}: Self-Correction for Predicate Identification

User: For the generated predicate identification result: predicate identification result., rethink the generated predicate identification result, evaluating the accuracy of the output predicate format, verify the correct identification of each predicate, and check for any missing predicates. Identify errors and make corrections accordingly. The predicate identification result format remains consistent with the previous format.
The output format example is: "Issues detected: ... Predicate identification result:". If no errors are found, output "Stop checking."

LLM: Result of self-correction.

Prompt $\mathcal{D}_k^2$: Argument labeling

. . . (**Prompt** $\mathcal{D}^1$)

User: In SRL, arguments refer to the components or phrases semantically related to a given predicate. They further describe the entities, actions, or concepts associated with the predicate in the sentence.Arguments are divided into core arguments and adjunct arguments.
The labels for all adjunct arguments are as follows:
EXT: extent
LOC: location
. . .
ARGA: secondary agent
PRR: predicating relation
Core arguments depend on the predicate, and a predicate may have different core argument frames. Within these frames, core arguments will have different interpretations.
Text: What @@was## the , purpose and goal of this campaign ? What are the arguments and their corresponding roles for the given predicate? The predicate is specified by @@ and ##.
For the predicate "was" in this text, it has the following frames:
For be as a verb:
Frame 1: The core arguments it has are: A1: topic, A1: comment.
Frame 2: The core arguments it has are: A1: thing that is.
By referring to the provided frames, determine the frame to which the predicate belongs in order to identify its core arguments. "R-" arguments are arguments that are referencing another argument in the sentence. "C-" arguments are discontinous spans that all refer to the same argument.
Rewrite the given text and enclose the beginning and end of the arguments with the corresponding <label> and </label> tags.

LLM: Argument labeling result.

Prompt C_{iter}^a: Self-Correction for Argument Labeling

User: For the generated argument labeling result: argument labeling result., check the generated argument labeling results for the following issues: whether the generated text is consistent with the original text, and whether the argument label correctly reflects the relationship between the predicate and the argument. Output any identified issues and correct them while maintaining the original argument annotation format. The output format should follow this example: "Issue detected: ... Argument labeling result: ". If no errors are found, output "Stop checking."

LLM: Result of self-correction.

Prompt: Generate Predicate Explanations

User: You are an expert in summarizing predicate meanings within a given frame. Your task is to infer and summarize the meaning of a predicate based on the provided Chinese predicate and frame arguments. AN represents argument labels in semantic role labeling tasks.
Here are some examples:
Predicate: abolish
Frame: A0: entity getting rid of, outlawing something; A1: thing abolished
Predicate meaning: get rid of, make illegal
Predicate: act
Frame: A0: agent; A1: predicate
Predicate meaning: play a role; behave
Predicate: act
Frame: A0: actor; A1: rounds for action
Predicate meaning: do something
Predicate: act
Frame: A0: actor, performer; A1: role, scenario enacted
Predicate meaning: perform a role
Predicate: predicate
Frame: frame
Predicate meaning:
Directly output the predicate meaning.

LLM: The meaning of the given predicate.