LLM-assisted Data Quality Improvement: SRL as a Case Study

Lijie Wang[*,1] **Yahui Liu**[*,2], **Yue Zhang**[3], **Zhenghua Li**[†,2]
[1]Applied Technology College of Soochow University, China
[2]Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, China
[3]School of Engineering, Westlake University, China
wlj4241@suda.edu.cn, yahuiliu.nlp@foxmail.com,
zhangyue@westlake.edu.cn, zhli13@suda.edu.cn

Abstract

Although large language models (LLMs) perform much worse than traditional supervised models on the semantic role labeling (SRL) task, our closer investigation reveals two important findings. First, the existing labeled SRL data is substantially flawed, i.e., of unsatisfactory quality. Second, LLMs can serve as an effective tool for detecting annotation errors and deciding which labels require human review, from the perspective of minimizing human effort. This paper proposes an LLM-assisted data quality improvement approach, and applies it to two SRL evaluation datasets from two different domains. With our approach, human annotators need to review only 11% of the originally annotated arguments flagged for manual inspection, among which 40% are truly erroneous. Meanwhile, our approach suggests 20% new arguments potentially missed in the original labels for human review, among which 60% are indeed missing. In total, our approach locates and corrects 21% errors in the two evaluation datasets. Experiments show that both traditional supervised models and LLMs perform much better on the revised evaluation datasets.

1 Introduction

Recently, large language models (LLMs) have shown remarkable generalization across various NLP tasks under the prompt-based framework, such as reasoning (Wei et al., 2022; Kojima et al., 2022), machine translation (Raunak et al., 2023) and text classification (Ouyang et al., 2022; Gilardi et al., 2023). The prompt-based framework takes natural language instructions, zero- or few-shot examples, and task-specific problems as input prompts, enabling LLMs to fulfill these prompts without additional fine-tuning. However, to the best of our knowledge, such prompt-based methods perform much worse than traditional supervised methods on linguistics-intense structural inference tasks, such as Semantic Role Labeling (SRL) and syntactic parsing (Cheng et al., 2024; Li et al., 2025; Zhang et al., 2025; Bai et al., 2025).

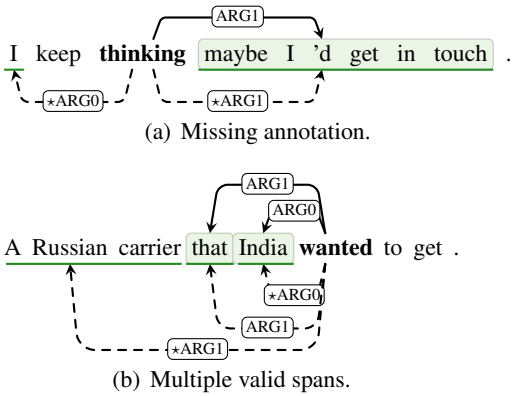


Figure 1: Examples of annotation quality issues in the original SRL test set. **Bold** words denote predicates; boxes and underlined text indicate argument spans; and labels (e.g., ARG0, ARG1) denote semantic roles. Solid arrows represent *original gold annotations* (above) , while dashed arrows indicate *corrected annotations* (below). The symbol ⋆ marks LLM-generated arguments.

This work focuses on SRL, a fundamental NLP task that aims to identify the predicate-argument structure of a sentence (Pradhan et al., 2005, 2012; He et al., 2015; Wang et al., 2022). Notably, Cheng et al. (2024) reported that the prompt-based ChatGPT (Achiam et al., 2023) achieves an F1 score of only 38.6% on the CoNLL-2012 SRL benchmark (Pradhan et al., 2012), whereas the traditional MRC-SRL model [1] (Wang et al., 2022) achieves 88.3%. Motivated by this discrepancy, we use SRL as a case study to investigate the performance and challenges of prompt-based LLMs on such tasks.

We reimplemented the work of Cheng et al. (2024) by employing the more sophisticated o1-

[*] Equal contribution.
[†] Corresponding author.

[1]MRC-SRL (Machine Reading Comprehension-based Semantic Role Labeling) formulates SRL as a question-answering task, using role-specific queries to extract argument spans.

mini (Jaech et al., 2024) for argument generation. Despite achieving an F1 score of 55.3%, its performance remains substantially inferior to that of traditional supervised models. Upon closer inspection of the LLM outputs, we observed that some predictions, though diverging from gold annotations, are semantically correct. As shown in Figure 1(a), the LLM correctly identifies "*I*" as the ARG0 of "*thinking*", while the gold annotation misses this argument. Further analysis of the discrepancies reveals that a significant proportion of these cases stem from errors in the gold annotations and limitations in current annotation guidelines, which overlook multiple valid spans for a single role. For instance, in Figure 1(b), the LLM generates a valid span "*A Russian carrier*" for the ARG1, whereas the gold annotation is simply "*that*". Consequently, valid LLM generations that recover missing annotations or provide alternative valid spans are misleadingly penalized as false negatives, distorting the assessment of LLM capabilities. Therefore, refining datasets and annotation guidelines is imperative for accurately evaluating LLMs in SRL and advancing the field.

Inspired by our finding that LLMs effectively capture semantic structures, we propose an LLM-assisted data quality improvement approach. First, we employ several LLMs to generate candidate spans for each semantic role, supplementing gold annotations by recovering missing roles and providing other valid spans for existing roles. Second, we use an LLM to assess the validity of each predicate-argument pair, where arguments are drawn from gold annotations, traditional supervised model predictions, and LLM generations. Finally, human review is prioritized for gold annotations flagged as invalid and for newly collected spans judged as valid. This strategy exhibits high efficiency: reviewing only 11% of the originally annotated arguments corrects 40% of erroneous labels, and suggesting an additional 20% of argument spans achieves a validity rate of 60%.

In summary, the main contributions of our work are as follows:

- We conduct the first systematic analysis of quality issues in mainstream SRL benchmarks. Accordingly, we refine annotation guidelines to accommodate multiple valid spans and optional annotations, and correct two SRL evaluation datasets from two different domains.
- We propose a simple yet effective data optimization framework that achieves high data quality through reduced inspection, locating and correcting 21% of the errors in the two evaluation datasets.
- On the revised test set, both traditional supervised methods and LLM-based approaches show significant performance improvements and robust cross-domain generalization, indicating that annotation deficiencies in existing benchmarks have underestimated model capabilities[2].

2 Related Work

2.1 SRL Modeling

Traditional supervised SRL methods generally fall into three categories: **transition-based**, **graph-based**, and **generative** methods. Transition-based methods cast SRL as an incremental decoding process, employing a sequence of actions to construct predicate-argument structures with explicit structural constraints (Fernández-González and Gómez-Rodríguez, 2020; Fei et al., 2021). Graph-based methods model the structured dependencies between predicates and arguments, covering both end-to-end (He et al., 2018; Cai et al., 2018; Fei et al., 2020) and given-predicate settings (He et al., 2017; Zhou et al., 2022; Zhang et al., 2022). Generative methods reformulate SRL as a sequence-to-sequence generation task (Daza and Frank, 2018; Rexhina Blloshmi, 2021).

Recent studies have explored the application of LLMs to SRL through prompt-based generation. Cheng et al. (2024) proposed Prompt-SRL, a method that outperforms supervised models in the same few-shot settings (e.g., 3-shot) but still lags behind traditional models trained on full datasets. Li et al. (2025) further employed retrieval-augmented generation and self-correction to annotate predicate-argument pairs for fine-tuning an LLM, achieving state-of-the-art performance in end-to-end settings and matching traditional methods in given-predicate settings.

Despite these advances, existing studies have primarily focused on improving model performance through various strategies, while paying little attention to data quality, particularly the quality of evaluation benchmarks, where erroneous annotations can underestimate model capabilities.

[2]Our code, revised dataset, and evaluation protocols are available at https://github.com/yahui19960717/LLM-srl-data-quality for research purposes.

Issue Type	Sentence	Role:Desc.	Span Annotation Type		
			Original	LLM-G	Corrected
Boundary	And I *got* a card from.	ARG2:giver	"from" → ✗	None	None
Missing	Among other things, the US has worked with Bangladesh to *establish* ...	ARG0:agent	None	"the US" → ✓	"the US" and "Bangladesh"
Multiple valid spans	I think there are a number of other measures that *indicate* how well somebody's doing running a company ...	ARG0:indicator	"of other measures" → ✗	"a number of other measures" → ✓	"other measures" or "a number of other measures"
Optional	and she's telling me about you *know* her sister's wedding.	ARG0:knower	"you" → ✓	"you" → ✓	"you" (optional)

Table 1: Examples of four types of annotation quality issues. Each example consists of a sentence, a predicate (highlighted in red italics), and a role label along with its description (Role: Desc.). "Original" denotes gold annotations, "LLM-G" denotes the argument spans generated by an LLM, and "Corrected" denotes the revised annotations. The symbols ✓/✗indicate whether the spans are judged as correct by an LLM.

2.2 Annotation Error Detection Methods

Recent studies have shown that annotation errors are pervasive in widely used NLP benchmarks (Northcutt et al., 2021). Existing annotation error detection (AED) work have mainly focused on tasks such as part-of-speech tagging (Liu et al., 2025b) and named entity recognition (Reiss et al., 2020), while SRL has received comparatively limited attention. The most relevant prior work for SRL, Dickinson and Lee (2008), introduces a variation-based method for detecting local predicate-argument inconsistencies but does not address dataset-level issues. This limitation is further reflected in Klie et al. (2023), who systematically evaluate 18 AED methods across text classification, token labeling, and span labeling, and observe that structured semantic tasks such as SRL remain underexplored.

To bridge this gap, we conduct a thorough investigation into data quality issues in SRL and demonstrate that rectifying annotation errors leads to more reliable evaluation and greater performance gains across a range of architectures.

3 An Analysis of Annotation Quality Issues in SRL Test Sets

To investigate the inferior performance of LLMs compared to traditional supervised models on SRL, we employed an LLM to generate predicate-argument structures for sampled instances from CoNLL-2012 (Pradhan et al., 2012) and conducted a fine-grained error analysis. Specifically, given a tuple of {*sentence, predicate, roles, role descriptions*[3]}, the LLM was prompted to extract a corresponding argument span from the sentence for each role. The prompt used to guide this extraction process is shown in Figure 4 (Appendix A). Then we performed string matching to compare the LLM-generated span against the gold annotated span for each predicate-role pair. In cases of discrepancy, two analysts independently labeled the underlying causes, such as LLM hallucination, semantic misunderstanding by the LLM, or errors in original gold annotations. Our analysis of 100 such cases revealed that nearly half of the LLM-generated outputs were reasonable, primarily attributable to errors or omissions in the original annotations.

We subsequently categorized these annotation quality issues into four primary types, with corresponding examples shown in Table 1 (see Table 4 in Appendix B for additional examples):

- **Boundary Error**: this error type refers to instances where an annotated argument exhibits imprecise span boundaries or is spurious (i.e., the argument is non-existent).
- **Missing Annotation**: this error type refers to instances where a valid argument span exists in the sentence but was omitted from the gold annotations.
- **Multiple Valid Spans**: this category refers to cases where multiple valid spans exist for a role in a sentence, but the gold annotations include only one of them.

[3]The *role description* is derived from the frame file of PropBank (https://propbank.github.io/v3.4.0/frames), which defines the frames of the predicates and provides descriptions for their corresponding core arguments.

Data Source	LLM-J	Valid Rate (%)
Golden data	Valid	93.3–100
	Invalid	60.0
Collected new data	Valid	43.3–66.7
	Invalid	6.7

Table 2: Manual spot-check results of LLM judgments (LLM-J) on gold and newly collected data. *Valid Rate* denotes the proportion of valid data verified by human annotators, reported as a range across the two test sets.

- **Optional Annotation**: this category refers to instances where, for non-essential predicates (e.g., colloquial parentheticals), incomprehensible sentences, or ambiguous role descriptions, the annotation of corresponding arguments is often inconsistent in current datasets. Consequently, such arguments should be designated as optional.

The first two types correspond to conventional annotation errors, while the latter two types mainly stem from limitations in the current annotation guidelines, as they overlook scenarios involving multiple valid spans for a single role or inherently optional annotation decisions.

Notably, such data quality issues not only skew the evaluation of LLMs' capability on SRL but also hinder progress in the field of SRL by providing unreliable signals for model comparison and development. This motivates us to revise test sets and refine the corresponding annotation guidelines.

4 Our Method

4.1 Exploring LLM Assistance in SRL Annotation

Our prior analysis in Section 3 demonstrates that LLMs are indeed capable of capturing structured semantics. Building on this insight, we further investigate their potential to assist in the manual annotation process and conduct an empirical study to assess their effectiveness.

First, we employed three LLMs (i.e., o1-mini (Jaech et al., 2024), DeepSeek-V3.2 in thinking mode (Liu et al., 2025a), and GPT-4o (Hurst et al., 2024)) to generate argument spans for missing roles and alternative valid spans for existing roles (i.e., "*Missing Annotation*" and "*Multiple Valid Spans*" in Table 1), which serve as a supplement to gold annotations. Specifically, given the tuple of {*sentence, predicate, role, role description*}, the LLM generated an argument span for each semantic role using the prompt shown in Figure 5 (Appendix A). Human evaluation on 50 sampled instances shows that by aggregating the gold annotations, predictions from traditional supervised models, and the LLM-generated outputs, the recall of valid spans across all predicate-role pairs reaches 97.9%, with LLM outputs contributing an additional 40.7% to the newly collected arguments. These results indicate that LLMs can effectively supplement missing annotations or spans in the golden data.

Second, to handle large-scale data requiring manual verification, we employed LLMs to identify samples for manual review and annotation, i.e., erroneous instances in gold annotations and valid arguments in newly collected data. Specifically, given a tuple of {*sentence, predicate, role description, argument span*}, we prompted the LLM to assess the validity of the predicate-argument pair via the prompt in Figure 6 (Appendix A). In a preliminary investigation, we assessed several LLMs as judges, including o1-mini, DeepSeek-V3.2, and Qwen3-Max (Yang et al., 2025). Following a comprehensive cost–quality trade-off analysis, we adopted o1-mini as the sole judge model. Details regarding this selection are provided in Appendix C. Subsequently, we randomly sampled instances from both the valid and invalid sets to manually evaluate the actual proportion of valid data.

For the gold annotations, among instances deemed valid by LLM-J, we conducted random spot checks on 30 samples from each of the two test sets. The proportion of valid spans in one set was 93.3% (reflecting the accuracy of gold annotations), with 6.7% of cases exhibiting multiple valid spans; conversely, the proportion in the other set was 100%, with no multiple spans observed. In contrast, for instances classified as invalid by LLM-J, the proportion of valid spans in the randomly spot-checked subsets of both test sets was approximately 60%.

For the newly collected arguments, we randomly sampled 30 instances from each of the two sets judged as valid by LLM-J, revealing validity rates of 43.3% and 66.7%, respectively. Conversely, among 30 randomly sampled instances judged as invalid by LLM-J from the same two sets, the proportions of valid data were only 6.7% in both sets.

The manual spot-check results are detailed in Table 2, demonstrating the potential of leveraging LLMs to uncover annotation errors in gold data and identify valid arguments in new data.

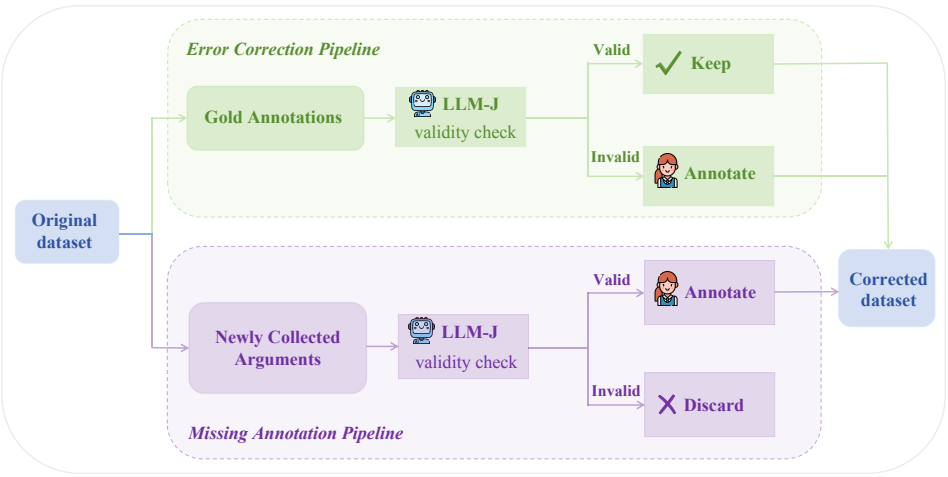


Figure 2: Data processing and annotation workflow. The *Error Correction Pipeline* identifies and corrects erroneous annotations in the original test set; the *Missing Annotation Pipeline* identifies and completes missing annotations in the gold annotations. *LLM-J* denotes the LLM's judgement, which is detailed in Section 4.1.

Therefore, to improve efficiency and reduce manual workload in dataset correction, we propose an LLM-assisted data optimization approach. By leveraging LLMs for span generation and candidate data selection, this approach effectively reduces manual annotation costs while ensuring high precision and recall in role annotation. To provide a comparison with existing AED methods, we re-implemented a previously proposed AED method based on marginal probabilities and reported the results in Appendix D.

4.2 Data Processing and Annotation Workflow

The overall workflow is illustrated in Figure 2. It consists of two main pipelines: an error correction pipeline designed to revise existing gold annotations, and a missing annotation pipeline aimed at collecting and verifying additional valid predicate-argument pairs. The following paragraphs provide a detailed introduction of each component.

Original Dataset We curated data from the CoNLL-2012 dataset (Pradhan et al., 2012), a standard benchmark for span-based SRL. To assess cross-domain performance, we selected the BN (broadcast news) and TC (telephone conversation) subsets as near- and far-domain test sets. We then randomly sampled 500 sentences from each subset, excluding those without predicates or containing only *be* verbs. Furthermore, we retained only core arguments and excluded non-core arguments, as they often lack concrete semantics or have ambiguous definitions, resulting in low annotation consistency[4] and posing challenges for non-linguists (He et al., 2015; Cheng et al., 2024).

Error Correction Pipeline This pipeline is designed to identify and correct erroneous instances in gold annotations. Specifically, an LLM (i.e., o1-mini) is prompted to assess the validity of each predicate-argument pair in the gold annotations. Based on the pilot validation in Table 2, gold annotations judged as valid by LLM-J are retained as low-priority cases for manual revision and directly incorporated into the corrected test set, while those judged as invalid are sent to human annotators for correction. The corrected outputs are then integrated into the revised test set.

Missing Annotation Pipeline We employ a dual strategy to identify potentially missing arguments and other valid spans for each predicate-role pair: aggregating predictions from multiple traditional supervised models and collecting generations from three LLMs, subsequently filtering out spans already present in the gold annotations. For these newly collected predicate-argument pairs, the LLM-J is further employed to evaluate the validity of each pair. Guided by the pilot validation results in Table 2, we exclude candidates deemed invalid by LLM-J from manual verification to reduce annotation costs, while forwarding those classified

[4] He et al. (2015) proposed a QA-based annotation method, and their evaluation of non-core arguments (e.g., *adverbial modifiers*, *extent*) against the gold-standard CoNLL-2009 yielded a precision of 59.9% and a recall of 63.6%.

as valid to human annotators for verification and correction. The verified arguments are then incorporated into the revised test set.

Corrected Dataset The final revised test set comprises three components: retained gold annotations, human-corrected gold annotations and newly collected arguments verified through manual review.

Agreement between LLM-J and Human Annotators We calculated the kappa statistic (Cohen, 1968) to measure the agreement between LLM-J and human judgments. For instances not requiring human review (i.e., original annotations judged as valid by LLM-J and newly collected data judged as invalid), LLM-J and human judgments achieve a Cohen's kappa of 0.89. This high agreement demonstrates the reliability of the automated decisions in these cases. Conversely, for data flagged for human review (i.e., original annotations judged as invalid by LLM-J and newly collected data judged as valid), the kappa value is 0.23. This lower agreement indicates that LLM-J struggles with these ambiguous cases, thereby validating the necessity of human intervention.

4.3 Refinement of Annotation Guidelines

Motivated by the annotation quality issues identified in Section 3, we refined the annotation guidelines to address two scenarios not adequately covered by the original protocols: (i) the existence of multiple valid spans for a single role, and (ii) the optionality of annotations in cases where it is ambiguous whether an argument should be annotated.

Multiple Valid Spans Our analysis revealed that a single argument may correspond to multiple valid spans in a sentence. For instance, in the sentence "*Barak , who has been forced to call national elections, offered to recognize a Palestinian state*", the "*entity offering* " for the predicate "*offered*" could reasonably be annotated as either "*Barak*" or the full noun phrase "*Barak, who has been forced to call national elections*". This inconsistency frequently arises concerning the inclusion of modifiers, restrictive relative clauses, and appositives. Given that the existing dataset exhibits inconsistent annotations for such cases and the original guidelines lack clear specifications regarding their inclusion, we revise the guidelines to annotate all semantically valid spans for a role.

Optional Annotation We identify three specific cases where the annotation is labeled as optional:

- **Ambiguous role definition.** Consider the predicate "*based*" in the sentence "*All this happening about 90 minutes ago based on the wire reports we 're getting here*". The description for the role "ARG1" is "*decision, whole*", and the gold annotation is "*All this happening about 90 minutes ago*". Due to the ambiguity of the role description and the resulting low inter-annotator agreement, this role is annotated as optional.
- **Ungrammatical or incomprehensible sentence.** For example, in the sentence "*And there were having a sale on the wine dad wanted*", the "*owner*" for the predicate "*having*" is annotated as "*there*" in the gold annotations. Since the sentence contains a grammatical error (likely a typo), arguments in such contexts should ideally be left unannotated, and we labeled them as optional.
- **Colloquial parentheticals.** For instance, in the utterance "*and she 's telling me about you know her sister 's wedding*", the predicate "*know*" functions as a colloquial parenthetical, rendering its semantic roles less critical. So these roles are annotated as optional.

Accordingly, we revise the evaluation metric as follows: (1) for roles with multiple valid spans, a prediction is considered correct if it matches any of the ground truth annotations; and (2) for optional annotations, a match between the prediction and the ground truth is counted as correct, while the absence of a prediction does not penalize the model's recall. The detailed evaluation strategy is presented in Algorithm 1.

4.4 Manual Annotation Mechanism

For the annotation process, we employed three annotators with expertise in syntactic and semantic parsing, all of whom possessed a solid understanding of English and the SRL task. First, the annotators were provided with a brief tutorial and a representative set of sample annotations drawn from the frame files. Then, given a predicate-role pair and the corresponding context, the annotators were asked to check its validity, revise the provided span, and identify additional valid spans. The annotation interface is shown in Figure 7 (Appendix E). On average, verifying and revising a predicate-role pair required approximately 2 minutes for the TC set and 6 minutes for the BN set, depending on sentence length and complexity. The average inter-annotator agreement rate was 78.7%.

Algorithm 1 Core Computation for Precision, Recall, and F1 Metrics on the Revised Set.

```
Metrics ← Dict{TP : 0, FP : 0, FN : 0}
AllRoles ← Gold.roles ∪ Pred.roles
for role ∈ AllRoles do
    GoldSpans ← Gold.roles.spans
    PredSpan ← Pred.roles.spans
    if GoldSpans ≠ ∅ AND PredSpan ≠ ∅ then
        ▷ A prediction is considered correct if it matches a candidate
        if PredSpan ∈ GoldSpans then
            Metrics[role][TP] += 1
        else
            Metrics[role][FP] += 1
            Metrics[role][FN] += 1
        end if
    else if GoldSpans ≠ ∅ then
        ▷ If the role is optional, failing to identify it does not penalize recall
        if role is not Optional then
            Metrics[role][FN] += 1
        end if
    else if PredSpan ≠ ∅ then
        Metrics[role][FP] += 1
    end if
end for
```

During the review process, if the results from the three annotators were consistent, their output was adopted directly. In cases of discrepancy, a consensus discussion was convened involving all annotators and two additional reviewers to resolve the differences.

4.5 Post-Correction Statistics

Figure 3 reports the rates of the three error types for the selected BN and TC subsets. The two subsets originally contain 2,132 and 1,844 argument spans, respectively. Based on LLM judgement, 11.2% and 11.0% of these spans were flagged for manual review, within which we corrected 40.6% and 39.2% of the label errors, respectively.

Additionally, among the newly collected spans (22.9% for BN and 17.5% for TC), 44.8% (BN) and 67.6% (TC) were confirmed as missing annotations, while 15.8% (BN) and 12.3% (TC) were confirmed as valid supplementary spans[5] for existing roles.

[5]Note that the multiple valid spans contributing to the rate in Figure 3 includes both model-generated spans and those added manually.

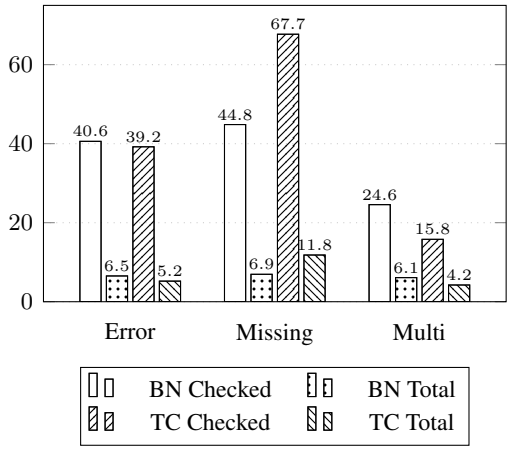


Figure 3: Error, missing, and multiple candidate rates (%) of the original annotations. Solid bars: rates over manually checked instances (*Checked*); hatched bars: rates over the entire dataset (*Total*).

Overall, approximately 60% of the newly collected spans are valid (see Appendix G for computation details).

In total, we corrected 21.1% of errors in the BN set and 21.3% in the TC set, covering all three error types shown in Figure 3. Finally, the revised BN set contains 2,425 argument spans for 2,280 core roles, while the revised TC set contains 2,140 argument spans for 2,062 core roles.

5 Experiments

5.1 Experimental Setup

Datasets. To investigate cross-domain generalization, we conduct experiments on the CoNLL-12 English SRL benchmark (Pradhan et al., 2012), a multi-domain dataset encompassing seven genres: newswire (NW), broadcast news (BN), broadcast conversation (BC), magazine (MZ), telephone conversation (TC), web data(WB) or pivot text (PT) (Pradhan et al., 2013). In this study, models are trained on the NW domain and evaluated on the BN and TC domains. The BN domain exhibits stylistic similarity to NW, as both are characterized by relatively formal and well-structured text. In contrast, TC consists of conversational speech featuring colloquialisms and disfluencies, thereby representing a more substantial domain shift. This setup facilitates a systematic comparison of model behavior under near-domain and far-domain conditions.

All experiments in this work are conducted under the predicate-given setting, with a focus on the labeling of core arguments.

Model	Test Set	BN			TC		
		P	R	F1	P	R	F1
SemiCRF	Original	84.99	84.71	84.85	79.22	85.99	82.47
	Corrected	90.12	86.67	88.36	90.08	89.63	89.86
	Δ	*+5.13*	*+1.96*	*+3.51*	*+10.86*	*+3.64*	*+7.39*
TreeCRF	Original	85.81	87.38	86.59	79.17	87.34	83.05
	Corrected	90.60	88.82	**89.70**	91.02	91.60	**91.31**
	Δ	*+4.79*	*+1.44*	*+3.11*	*+11.85*	*+4.26*	*+8.26*
LLM-G (o1-mini)	Original	61.55	69.84	65.44	62.06	76.46	68.51
	Corrected	66.27	73.23	69.57	68.66	78.51	73.26
	Δ	*+4.72*	*+3.39*	*+4.13*	*+6.60*	*+2.05*	*+4.75*
LLM-G (DeepSeek)	Original	53.09	58.77	55.79	54.79	63.94	59.01
	Corrected	56.04	60.50	58.19	60.39	65.41	62.80
	Δ	*+2.95*	*+1.73*	*+2.40*	*+5.60*	*+1.47*	*+3.79*
LLM-G (GPT-4.1)	Original	60.88	68.76	64.58	64.51	72.44	68.25
	Corrected	64.45	70.68	67.42	68.40	76.68	72.30
	Δ	*+3.57*	*+1.92*	*+2.84*	*+3.89*	*+4.24*	*+4.05*

Table 3: Overall Precision (P), Recall (R), and F1 (%) on the original and corrected test sets for the BN and TC domains of CoNLL12. Δ denotes the performance gain achieved by correcting the test set. LLM-G models are evaluated under the loose-match metric, and strict-match results are shown in Table 6 (Appendix H).

Models. We evaluate both traditional supervised models and LLM-based models on the original and corresponding corrected test sets. For traditional supervised models, we assess two representative span-based methods: SemiCRF (Lin et al., 2021) and TreeCRF (Zhang et al., 2022), both of which are trained on NW training data. For training-free LLM-based models (LLM-G), we follow the methodology of Cheng et al. (2024), who prompted ChatGPT (Achiam et al., 2023) to generate all label-span pairs for a given predicate in a single pass. In our work, we employ o1-mini, DeepSeek-V3.2 (in non-thinking mode), and GPT-4.1 (Achiam et al., 2023) as the generative models.

Evaluation Metrics. We adopt standard span-level SRL metrics, namely Precision (P), Recall (R), and F1 score. To accommodate two specific annotation refinements, i.e., multiple valid spans and optional annotations, we have accordingly adapted the metric computation, as detailed in Algorithm 1.

For traditional supervised models that predict the start and end indices of argument spans, we evaluate performance by comparing the predicted indices with the gold annotations. In contrast, for LLM-based models that generate argument spans as text, we assess prediction correctness via exact string matching against the ground-truth spans. However, preposition inclusion within argument spans significantly affects matching results. For example, given the predicate *questions* in the sentence "*We begin tonight with urgent questions for the government*", the gold annotation for the role "*entity asked*" is "*for the government*", whereas an LLM-G may generate "*the government*". To address this, we relax the matching criteria by ignoring initial prepositions and report results under both strict and loose matching criteria.

5.2 Main Results

Table 3 details the Precision, Recall, and F1 scores for both traditional supervised models and the LLM-G models. We observe that SemiCRF, TreeCRF, LLM-G (o1-mini), LLM-G (DeepSeek), and LLM-G (GPT-4.1) achieved F1 score improvements of 3.51, 3.11, 4.13, 2.40, and 2.84 on the BN set, and 7.39, 8.26, 4.75, 3.79, and 4.05 on the TC set, respectively. These results indicate a consistent performance boost across both traditional and LLM models on the new test sets, implying that annotation errors in the original test sets constrained the models' capabilities. Then we discuss several key findings.

First, the improvement in model performance aligns with the proportion of error corrections in the dataset. Both traditional models and LLM-based models exhibited greater improvements in precision compared to recall. For instance, on the BN set, SemiCRF, TreeCRF, and LLM-G (o1-mini)

achieved precision gains of 5.13, 4.79, and 4.72, respectively, versus recall gains of 1.96, 1.44, and 3.39. This disparity in improvement is largely attributable to the nature of annotation errors: while recall is primarily affected by boundary errors and multiple valid spans, precision is more sensitive to boundary errors and missing annotations (false negatives). As indicated in Figure 3, the proportions of annotation errors and missing annotations in two sets are higher than those of multiple valid spans, thereby resulting in more substantial gains in precision.

Second, correcting the test sets proved crucial for accurately evaluating the cross-domain performance of traditional supervised models. On the original test sets, models trained on the NW domain naturally performed better on the near-domain (BN) than on the far-domain (TC). However, on the revised test sets, this performance gap narrows, with the TC domain even surpassing BN. This suggests that the cross-domain generalization capability of traditional models is robust but was previously obscured by annotation errors. Specifically, in the original TC test set, the presence of colloquial expressions led to missing annotations for corresponding predicates. Consequently, correct predictions were treated as false negatives, artificially depressing the accuracy scores. On the corrected test data, performance on the TC set exceeds that of the BN set, primarily due to the relative simplicity of the TC data.

Finally, LLM-based methods exhibit robustness to domain variations but sensitivity to task difficulty. All three LLM-G models demonstrated consistent performance gains across both test sets. This stability is primarily attributed to their training-free prompting paradigm, which renders them inherently less susceptible to domain shifts than supervised approaches. Meanwhile, the consistent improvements observed across all three LLM-G models suggest that using o1-mini as the judge model does not introduce a favorable evaluation bias toward LLM-G (o1-mini). Moreover, all models achieved higher performance on the TC test set than on BN, as TC is considerably simpler. For instance, TC contains an average of 1.3 predicates per sentence, compared to 3.1 in BN.

6 Conclusion

In this study, we conduct a systematic diagnosis of LLM performance on SRL and identify substantial annotation quality issues in existing SRL test sets. In particular, our manual inspection shows that nearly half of the discrepancies between LLM outputs and gold annotations stem from annotation quality issues, which can obscure the accurate evaluation of model capabilities.

To efficiently mitigate these data quality issues, we introduce a simple yet effective framework for data optimization. With this framework, human annotators correct 40% of erroneous labels by reviewing only 11% of the original gold annotations. Furthermore, our approach suggests an additional 20% of candidate arguments for missing roles and spans, of which 60% are confirmed as valid. Based on corrected test sets, our experiments show that both traditional supervised models and LLM-based generative models achieve notable performance improvements, with traditional models exhibiting robust cross-domain performance.

Limitations

First, our scope is currently limited to verifying and correcting core roles in the BN and TC sets. Optimizing semantically distinct non-core roles and extending our LLM-assisted method to other domains remain directions for future work. Second, although the proposed LLM-assisted framework effectively reduces labor costs, approximately 6.7% of the data validated by the LLM still contains errors. Third, the LLM generation method used in this study is only one instance of applying LLMs to SRL and does not fully reflect their potential. Nevertheless, the insights from our analysis provide a solid foundation for future research.

Acknowledgments

The authors would like to thank the anonymous reviewers for the helpful comments. We are greatly grateful to all our annotators for their hard work in data annotation. This work was supported by National Natural Science Foundation of China (Grant No. 62576237 and 62336006), and a Project Funded by the Priority Academic Program Development (PAPD) of Jiangsu Higher Education Institutions.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman,

Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Xuefeng Bai, Jialong Wu, Yulong Chen, Zhongqing Wang, Kehai Chen, Min Zhang, and Yue Zhang. 2025. Constituency parsing using llms. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3762–3775.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2753–2765.

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 *International Conference on Intelligent Computing*, pages 50–61.

Jacob Cohen. 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213–220.

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.

Markus Dickinson and Chong Min Lee. 2008. Detecting errors in semantic annotation. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, pages 605–610.

Hao Fei, Yafeng Ren, and Donghong Ji. 2020. High-order refining for end-to-end chinese semantic role labeling. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 100–105.

Hao Fei, Meishan Zhang, Bobo Li, and Donghong Ji. 2021. End-to-end semantic role labeling with neural transition-based model. In *Proceedings of the AAAI conference on artificial intelligence*, pages 12803–12811.

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.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120.

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.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 643–653.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. abs/2410.21276.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jan-Christoph Klie, Bonnie Webber, and Iryna Gurevych. 2023. Annotation error detection: Analyzing the past and present for a more coherent future. *Computational Linguistics*, 49(1):157–198.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Xinxin Li, Huiyao Chen, Chengjun Liu, Jing Li, Meishan Zhang, Jun Yu, and Min Zhang. 2025. Llms can also do well! breaking barriers in semantic role labeling via large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 23162–23180.

ZhiChao Lin, Yueheng Sun, and Meishan Zhang. 2021. A graph-based neural model for end-to-end frame semantic parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3864–3874.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *CoRR*, abs/2512.02556.

Yahui Liu, Zhenghua Li, Chen Gong, Shilin Zhou, and Min Zhang. 2025b. Annotation error detection in painstakingly annotated data: Part-of-speech tagging as a case study. *Expert Systems with Applications*, 290:128374.

Curtis Northcutt, Anish Athalye, and Jonas Mueller. 2021. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Proceedings of*

the Neural Information Processing Systems Track on Datasets and Benchmarks, volume 1, pages 1–13.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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 H Martin, and Dan Jurafsky. 2005. Semantic role labeling using different syntactic views. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 581–588.

Vikas Raunak, Amr Sharaf, Yiren Wang, Hany Awadalla, and Arul Menezes. 2023. Leveraging GPT-4 for automatic translation post-editing. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12009–12024.

Frederick Reiss, Hong Xu, Bryan Cutler, Karthik Muthuraman, and Zachary Eichenberger. 2020. Identifying incorrect labels in the CoNLL-2003 corpus. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 215–226.

Rocco Tripodi Roberto Navigli Rexhina Blloshmi, Simone Conia. 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.

Nan Wang, Jiwei Li, Yuxian Meng, Xiaofei Sun, Han Qiu, Ziyao Wang, Guoyin Wang, and Jun He. 2022. An MRC framework for semantic role labeling. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2188–2198.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

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.

Ziyan Zhang, Yang Hou, Chen Gong, and Zhenghua Li. 2025. Self-correction makes LLMs better parsers. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 6749–6762.

Shilin Zhou, Qingrong Xia, Zhenghua Li, Yu Zhang, Yu Hong, and Min Zhang. 2022. Fast and accurate end-to-end span-based semantic role labeling as word-based graph parsing. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4160–4171.

A LLM Prompts

LLM Prompt to Generate Argument for a Given Predicate

Give a sentence: <*sentence*>, a predicate: <*predicate*>. all possible argument roles and their descriptions: <{*role: description*}>, please label the arguments about predicate <*predicate*> of the sentence. Provide the answer in JSON format as follows: {<*predicate*>: {*role*: *argument*}}.

Figure 4: The prompt from Cheng et al. (2024).

LLM Prompt to Generate Span for a Given Predicate-Role

The sentence: <*sentence*>
The predicate: <*predicate*>
All possible argument roles and their descriptions: <{*role: description*}>
The annotation role and its description: <*role*>, <*role-des*>
Please label the argument from the sentence that represents <*role-des*> for predicate <*predicate*>. Provide the answer in JSON format as follows: {{*role*: *argument*}}.

Figure 5: The prompt used for LLMs to generate argument spans.

Figure 4 illustrates the prompt introduced by Cheng et al. (2024), which served as the reference model in our preliminary study (Section 3) and as the baseline for our LLM-based methods (Section 5). Figure 5 depicts the prompt designed to generate candidate spans for predicate-argument pairs, employed in Section 4.1 to supplement gold annotations. Finally, Figure 6 presents the prompt used for LLM-based judgment to assess the validity of predicate-argument pairs.

B Quality Issues in Gold Annotations

We provide additional examples to better illustrate the data quality issues, as shown in Table 4.

C Preliminary Assessment and Selection of LLM Judges

For argument validity judgment, we evaluated o1-mini, DeepSeek-V3.2 (thinking and non-thinking modes), and Qwen3-Max on the BN test set. Our analysis revealed two key findings. First, among instances flagged as invalid by o1-mini but judged as valid by at least one other model, manual verification revealed that only 33.9% were actually erroneous. Consequently, all instances flagged as invalid by o1-mini were sent to mandatory human review. Second, for instances judged as valid by o1-mini, we investigated whether DeepSeek-V3.2 and Qwen3-Max could identify genuine errors missed by o1-mini. However, these models detected only very few additional true errors, while producing many extra cases that would require human review. Thus, incorporating these models would substantially increase the human-review workload with minimal gains in detecting annotation errors. Based on this cost–quality trade-off, we retained o1-mini's valid judgments in the final pipeline and adopted o1-mini as the sole LLM judge. We leave the exploration of more effective multi-LLM judgment strategies to future work.

D Comparison with Probability-based AED Methods

Although annotation error detection (AED) has been extensively studied for structured prediction tasks such as part-of-speech tagging and named entity recognition (NER), no prior work has directly applied it to SRL. Given the structural similarity between NER and span-based SRL, we adapt a probability-based AED strategy to SRL as a baseline for comparison with our LLM-assisted framework. Specifically, we employ SemiCRF as the representative supervised baseline, as it provides candidate-level marginal probabilities that can be directly used for confidence-based filtering.

Table 5 presents the average argument validity rates and recall across different probability thresholds on two test sets. As the threshold rises, the validity rate of arguments above the threshold increases, while that of arguments below the threshold decreases. Notably, argument recall remains below 90% at all thresholds, whereas our LLM-assisted method achieves 97.9%. Furthermore, while the validity rate of the probability-based approach ranges from 90.27% to 91.67%, our method with LLM-based judgment attains an average validity rate of 96.65% (i.e., 93.3% and 100% on the two test sets, respectively). These results demonstrate that our LLM-assisted method outperforms conventional probability-based AED approaches.

E Annotation Interface

Figure 7 shows the interface for human review and annotation, where annotators verify one predicate-argument pair at a time. As shown in the right sidebar, the interface displays instance details: **Index**, **Full Sentence**, **Predicate**, role **Label** (e.g., ARG1), **Argument Meaning** (i.e., role description), and candidate argument spans (listed under **Annotate Argument**). In the sentence view, tokens

LLM Prompt for Judgement

You are a professional linguistic analysis expert and are now required to perform the semantic role labeling verification task. Your task is to evaluate the correctness of the argument and that of the semantic role label for the given sentence, predicate, argument and semantic role label. If any errors are found, you must provide a detailed explanation.

Fundamental Definition of SRL:
Semantic Role Labeling is a shallow semantic parsing task. Its core objective is to identify the arguments (i.e., participants or states) related to a given predicate and assign them specific semantic roles (e.g., Agent, Patient, Time, Location, etc.).
Predicate: refering to the word or phrase in a clause or sentence that expresses an action, behavior, state, relationship, or attribute.
Core Argument: refering to the word or phrase that plays a specific role in relation to the predicate, such as agent, patient, theme, instrument, benefactive, attribute, start point or end point, etc. Core arguments are essential components that help complete the meaning of the predicate and provide additional information about the event or state described by the predicate.

Execution Steps and Output Requirements:
1. Argument Extraction Verification:
- According to the given predicate and the phrase or clause where it is located, determine whether the given argument is correctly extracted based on syntactic and semantic knowledge. The possible results are: boundary correct, insufficient extraction, over extraction, boundary incorrect. That is to say, you need to judge whether the boundary of the given argument is correct, based on whether it expresses a complete meaning as the smallest semantic unit.
- Output in JSON format with the key fields being argument_extraction_evaluation and reason_for_argument_judgement. For the field of argument_extraction_evaluation, the candidate value is "yes" or "no".
2. Semantic Role Correctness Validation:
- Determine whether the argument and the predicate satisfy the specified role relationship. The primary results include: correct relation and incorrect relation. In this task, if the "{argument}" is the "{role_mean}" of the "{predicate}", then they satisfy the relation of "{role}", and the reverse is not true.
- Output in JSON format with the key fields being role_evaluation and reason_for_role_judgement. For the field of role_evaluation, the candidate value is "yes" or "no".
3. Summary of Results:
- According to the results of the argument extraction evaluation and the semantic role correctness validation, provide a summary judgement. If the fields of argument_extraction_evaluation and role_evaluation are both "yes", the final result is correct; if even one value is not "yes", the final result will be incorrect.
- Output in JSON format with the key fields being final_judgement and reason_for_final_judgement. The candidate values for final_judgement are "correct" and "incorrect".

Input Specification
We will input a sentence, a predicate, an argument, a role label and the description of this role label. Both the predicate and the argument are words or word fragments from the input sentence. Since there may be multiple identical words in the sentence but at different positions, to better distinguish which word is being referred to, the predicate in the input is marked with underscores (i.e., wrapped with _), and the semantic role field is highlighted in bold (i.e., wrapped with).

Example
Input:
Sentence: _Spending_ a lot of time hunting for bikes , and nah well . then getting one purchased , and then having to , you know , go get it and things like that .
Predicate: Spending
Argument: hunting for bikes
Role label: ARG2
Role description: in the clause or sentence where "Spending" is located, "hunting for bikes" represents the "instrument, benefactive, result, caused event, affected entity, topic, content, goal, destination, standard target, comparison target or attribute" of "Spending"
Output: {{"argument_extraction_evaluation": "yes", "reason_for_argument_judgement":"The argument "hunting for bikes" is a contiguous phrase in the sentence and forms a complete semantic unit.", "role_evaluation": "yes", "reason_for_role_judgement": "In the clause "Spending a lot of time hunting for bikes", the argument "hunting for bikes" specifies the purpose or goal of the predicate "Spending", which aligns with the role description for ARG2 as it includes goal or content.", "final_judgement": "correct", "reason_for_final_judgement": "Both the argument extraction and the semantic role labeling are correct."}}
Input:
Sentence: Which _was_ actually where the major problem came . Exactly . Exactly . yeah . Is that I suddenly had no transportation .
Predicate: was
Argument: Which
Role label: ARG1
Role description: in the clause or sentence where "was" is located, "Which" represents the "prototypical patient" of "was"
Output:{{"argument_extraction_evaluation": "yes", "reason_for_argument_judgement":"The argument "Which" is correctly extracted as a single word with clear boundaries, serving as a complete semantic unit as a relative pronoun in the sentence.", "role_evaluation": "no", "reason_for_role_judgement": " "Which" functions as the subject or theme, not as a prototypical patient.", "final_judgement": "incorrect", "reason_for_final_judgement": "The argument extraction is correct, but the semantic role label is incorrect because "Which" does not satisfy the specified role relationship as the prototypical patient of "was"."}}

Input
Sentence: {sentence}
Predicate: {predicate}
Argument: {argument}
Role label: {role}
Role description: in the clause or sentence where "{predicate}" is located, "{argument}" represents the "{role_mean}" of "{predicate}"
Please Answer only with a JSON object, without any additional text, explanations, or Markdown formatting.

Figure 6: The prompt used for LLM to assess the validity of the predicate-argument pair.

are indexed, the specific predicate is highlighted in *red*, and candidate spans are sourced from gold annotations and newly collected data.

The annotation workflow proceeds as follows: First, annotators perform a grammaticality check (*Sentence Check*); sentences deemed ungrammat-

Issue Type	Sentence	Role:Desc.	Span Annotation Type		
			Original	LLM-G	Corrected
Boundary	John M Wethy, ABC News, the Pentagon, still *looking* for answers.	ARG0:looker	"John M Wethy" → ✗	"John M Wethy, ABC News, the Pentagon" → ✓	"John M Wethy, ABC News, the Pentagon"
	I keep thinking maybe I 'd *get* in touch.	ARG1:thing	"I" → ✗	None	None
	I *think* Ozzy 's the guy that hit a head of a bat, you know ?	ARG1:thought	"Ozzy 's the guy that hit a head of a bat, you know" → ✗	"Ozzy 's the guy that hit a head of a bat" → ✓	"Ozzy 's the guy that hit a head of a bat"
Missing	I just *want* her to understand everything that has happened in the last several years .	ARG0:wanter	None	"I"→✓	"I"
	but it is a lot less expensive if we do it that way than if we actually make my parents in law *pay* it	ARG0:payer or buyer	None	"my parents" → ✗	"my parents in law"
	American warns it will seek more layoffs if it does *file* for Chapter 11 .	ARG1:file	None	"Chapter 11"→✓	"for Chapter 11" or "Chapter 11"
Multiple valid spans	Barak, who has been forced to call national elections, *offered* to recognize a Palestinian state.	ARG0:entity offering	Barak, who has been forced to call national elections → ✓	"Barak" → ✓	"Barak" or "Barak, who has been forced to call national elections"
	the most worthwhile thing as far as monetary value went that they *stole* was the pre-amplifier of my stereo	ARG1:thing stolen	"the most worthwhile thing" → ✓	"the pre-amplifier of my stereo" → ✓	"the most worthwhile thing" or "the pre-amplifier of my stereo"
	I think perhaps no one more surprised than some of the people *running* those firms on Wall Streat	ARG1:machine, operation, procedure	"those firms on Wall Streat" → ✓	"those firms" → ✓	"those firms" or "those firms on Wall Streat"
Optional	And there were *having* a sale on the wine dad wanted.	ARG0:owner	"there" → ✗	None	"there" (optional)
	But you *know* , I mean that mhm . should n't be a big deal .	ARG0:knower	you → ✓	you → ✓	"you" (optional)
	All this happening about 90 minutes ago *based* on the wire reports we 're getting here.	ARG1:decision, whole	"All this happening about 90 minutes ago" → ✗	None	"All this happening about 90 minutes ago" (optional)

Table 4: Extra examples for annotation quality issues. Each example consists of a sentence, a predicate (highlighted in red italics), and a role label along with its description (Role: Desc.). "Original" denotes gold annotations, "LLM-G" denotes the argument spans generated by an LLM, and "Corrected" denotes the revised annotations. The symbols ✓/✗indicate whether the spans are judged as correct by an LLM.

ical do not require further argument annotation. Next, annotators select valid spans from the provided candidates or manually add spans via *Custom Annotation*. For roles where annotation is optional, the annotator checks the *Optional Annotation* box, whereas the *No Argument* box is selected if no valid span exists. Finally, annotators click the *Submit* button to save the record to a local JSON file and advance to the next instance.

Additionally, the left sidebar facilitates manual saving and loading of annotations via dedicated buttons, while also displaying the annotation progress, including the total number of instances and the count of completed annotations.

F Experimental Implementation Details

F.1 Traditional Model Details

For TreeCRF, we directly used the open-source model from Zhang et al. (2022).

For our SemiCRF model for SRL, we adapt

Threshold	Recall (%)	Direction	Valid Rate (%)
0.50	89.15	↑	90.27
		↓	83.35
0.70	89.22	↑	90.95
		↓	59.69
0.90	88.77	↑	91.67
		↓	47.36
0.99	87.32	↑	91.62
		↓	41.86

Table 5: Validity rates and argument recall across different thresholds. ↑ and ↓ denote argument spans with probabilities above and below the threshold, respectively.

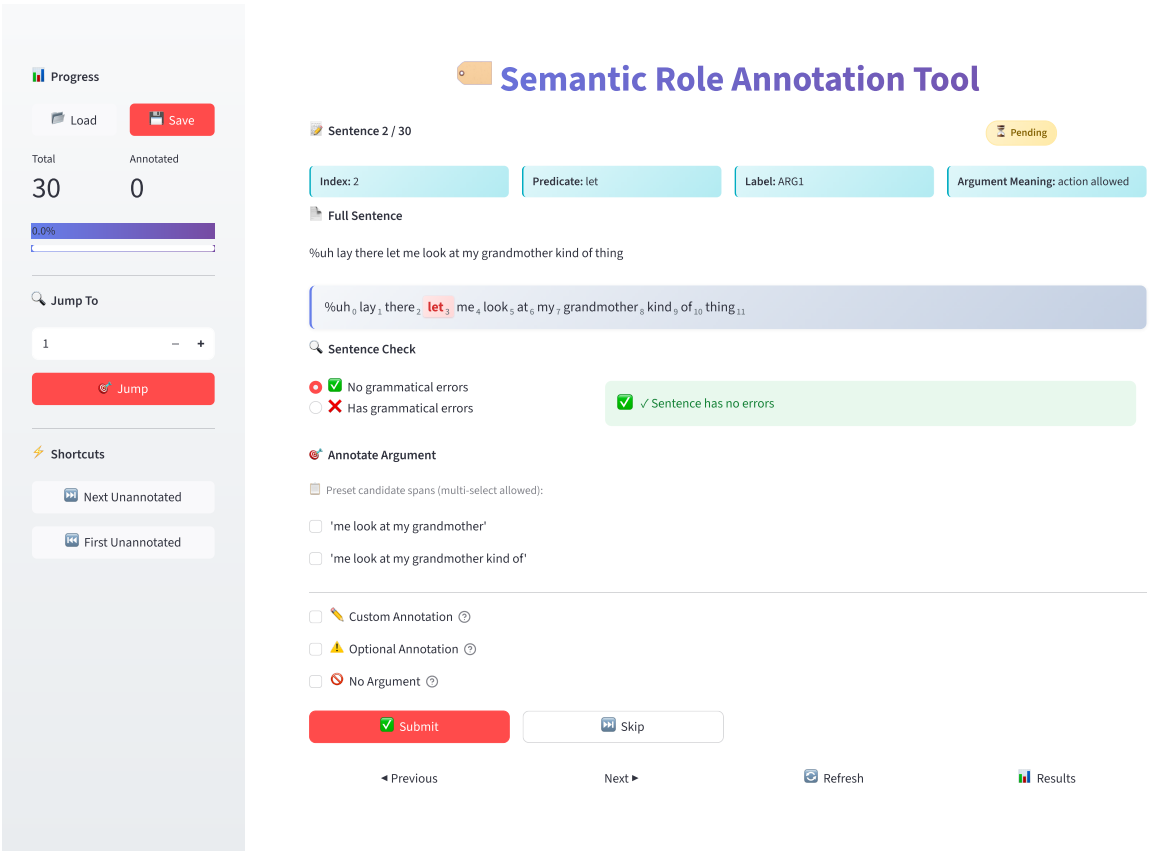


Figure 7: The annotation interface for human review and verification.

the network architecture described in Lin et al. (2021) with the following modifications. We used `bert-large-cased` as the pre-trained language model to encode the input sequence (a sentence and a given predicate). We encoded the given predicate by extracting its contextualized BERT representation and adding it to each token representation before span scoring. Then we applied two MLPs with a hidden size of 300 for span boundary prediction. Finally, a 300-dimensional biaffine scoring layer was applied to score candidate spans. In training, AdamW (Kingma, 2014) was used for optimization, with $\beta_1 = 0.9$, $\beta_2 = 0.9$, learning rate 2×10^{-5}, $\epsilon = 1 \times 10^{-12}$, and weight decay $\lambda = 0$. The model was trained for 80 epochs with early stopping (patience = 20).

F.2 Dataset Details

The traditional supervised models are trained on the NW domain, which comprises 15,288 sentences and 45,547 predicates. All models were evaluated

Model	Metric	Test Set	BN			TC		
			P	R	F1	P	R	F1
o1-mini	Strict-match	Original	54.77	62.15	58.23	58.71	72.34	64.82
		Corrected	59.45	65.13	62.16	64.70	73.98	69.03
		Δ	*+4.68*	*+2.98*	*+3.93*	*+5.99*	*+1.64*	*+4.21*
	Loose-match	Original	61.55	69.84	65.44	62.06	76.46	68.51
		Corrected	66.27	73.23	69.57	68.66	78.51	73.26
		Δ	*+4.72*	*+3.39*	*+4.13*	*+6.60*	*+2.05*	*+4.75*
DeepSeek-V3.2	Strict-match	Original	48.26	53.42	50.71	52.28	61.01	56.31
		Corrected	51.10	55.19	53.07	57.81	62.64	60.12
		Δ	*+2.84*	*+1.77*	*+2.36*	*+5.53*	*+1.63*	*+3.81*
	Loose-match	Original	53.09	58.77	55.79	54.79	63.94	59.01
		Corrected	56.04	60.50	58.19	60.39	65.41	62.80
		Δ	*+2.95*	*+1.73*	*+2.40*	*+5.60*	*+1.47*	*+3.79*
GPT-4.1	Strict-match	Original	54.19	61.21	57.49	58.46	70.66	63.98
		Corrected	57.97	63.74	60.72	61.69	74.57	67.52
		Δ	*+3.78*	*+2.53*	*+3.23*	*+3.23*	*+3.91*	*+3.54*
	Loose-match	Original	60.88	68.76	64.58	64.51	72.44	68.25
		Corrected	64.45	70.68	67.42	68.40	76.68	72.30
		Δ	*+3.57*	*+1.92*	*+2.84*	*+3.89*	*+4.24*	*+4.05*

Table 6: Precision (P), Recall (R), and F1 (%) of LLM-G models on the original and corrected BN/TC test sets under strict-match and loose-match metrics.

on the original and revised test sets for both BN and TC. Each set contains 500 sentences, with 1,321 and 1,281 predicates in BN and TC, respectively. The revised BN test set contains 2,425 argument spans for 2,280 core roles, whereas the original set contains 2,132 argument spans for 2,132 core roles. Similarly, the revised TC test set contains 2,140 argument spans for 2,062 core roles, compared to 1,844 spans for 1,844 core roles in the original set.

G Computation of Data Validity Ratio

The 60 validity rate is an argument-level metric based on newly collected arguments. Specifically, we collected 488 new arguments for the BN dataset (including 148 previously missed arguments and 79 alternative valid spans) and 322 for the TC dataset (comprising 218 missed arguments and 40 valid spans), yielding a rate of $\frac{148+79+218+40}{488+322} \approx 59.88\%$. Note that Figure 3 additionally includes manually added arguments, which are not counted in this calculation.

H LLM-G Evaluation Results under Strict-match

As shown in Table 6, the three LLM-G models achieve comparable performance gains under both strict and loose matching metrics, suggesting that the improvements are primarily attributable to enhanced dataset quality rather than the evaluation methodology.