Revisiting Semantic Role Labeling: Efficient Structured Inference with Dependency-Informed Analysis

Anonymous ACL submission

Abstract

Semantic Role Labeling (SRL) provides an explicit representation of predicate-argument structure, capturing linguistically grounded relations such as *who did what to whom*. While large language models (LLMs) have dominated recent NLP progress, these systems often rely on implicit semantic representations, lacking explicit structural constraints, systematic explanatory mechanisms, and reliability for complex downstream tasks. Traditionally, SRL systems have often relied on AllenNLP; however, the framework entered maintenance mode in December 2022, limiting compatibility with evolving encoder architectures and modern inference requirements. To address this, we revisit structured SRL modeling by introducing a modernized encoder-based framework that preserves explicit predicate-argument structure while enabling inference $10\times$ faster. Using modern encoders (BERT, RoBERTa, and DeBERTa), our framework maintains competitive performance while substantially improving inference efficiency. Furthermore, we adopt a dependency-informed diagnostic methodology to characterize span-level inconsistencies and analyze LLM behavior under dependency-informed structural signals. Finally, we illustrate the framework's downstream applicability through an English–French cross-lingual SRL projection case study.

1 Introduction

Semantic role labeling (SRL) is a fundamental task in Natural Language Processing (NLP) that identifies an explicit representation of predicate-argument structure, capturing linguistically grounded relations, i.e., *who did what to whom*. For example:

- [**ARG0** OpenAI] *released* [**ARG1** a new language model] [**ARGM-TMP** in March].
- [**ARG0** The research team] *evaluated* [**ARG1** its multilingual SRL performance] [**ARGM-LOC** on the Universal PropBank dataset].
- [**ARG1** The model] was *fine-tuned* [**ARG0** by engineers] [**ARGM-MNR** using dependency constraints].

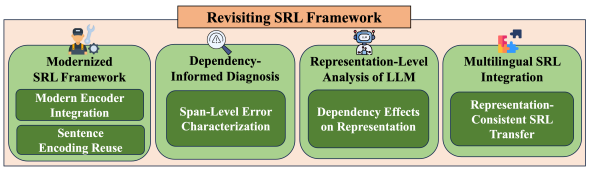

Figure 1: Overview of the revisiting SRL framework. Sentence-level encoding reuse enables efficient predicate-conditioned inferences across modern encoders. Dependency-informed diagnosis characterizes span-level errors, and representation-level analysis investigates dependency effects. The framework further supports representation-consistent cross-lingual SRL transfer as a downstream application.

Following Bonial et al. (2014), we assume a *predicate* to be a verb, noun, adverb, or adjective (e.g., ***afraid*** *of bears*), although we focus here on verbs (italicized above). Typical SRL labels are agent (ARG0), theme (ARG1), recipient or secondary argument (ARG2), and modifier (ARGM-TMP/LOC/MNR, etc.).

Due to its linguistic importance, SRL has long been a core NLP benchmark, appearing as shared tasks at CoNLL (Carreras and Màrquez, 2005; Hajič et al., 2009; Pradhan et al., 2012). This trajectory reflects broader shifts in modeling paradigms, from early statistical systems (He et al., 2017; Li et al., 2019) to span-based neural approaches using larger datasets, e.g., PropBank and FrameNet (Conia et al., 2022) and transformer encoders (Fei et al., 2021; Zhang et al., 2022).

The span-based AllenNLP SRL model (Gardner et al., 2018), based on Shi and Lin (2019), achieves high accuracy and supports various NLP tasks, including named entity recognition (NER), coreference resolution, and SRL. However, AllenNLP entered maintenance mode in December 2022, limiting compatibility with evolving encoder architectures and modern inference requirements. This obsolescence constrains the deployment of SRL in downstream tasks where predicate-argument features remain valuable—for example, stance detection (Mather et al., 2022), sentiment analysis (Klenner and Göhring, 2022), summarization (Fan et al.,

2023), and cross-lingual SRL projection (Youm et al., 2024). More broadly, they expose a gap between advances in semantic representation learning and the computational efficiency required for large-scale structured inference.

To overcome these integration barriers, we revisit structured SRL modeling by introducing a modernized encoder-based framework (see Figure 1). This design eliminates redundant per-predicate instance generation by caching sentence-level representations and reusing them across predicates, yielding 10× faster inference while preserving predictive performance. Using BERT-base, the model attains comparable performance (86.2%; AllenNLP: 86.72%), while RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2020) further improve predictive performance (87.07%, 87.49%). Across all evaluated encoder backbones, our framework remains faster than AllenNLP.

Beyond efficiency and predictive gains, the framework contributes (1) a dependency-informed diagnostic methodology for span boundary inconsistencies and structurally explainable deviations; (2) a representation-level analysis of how dependency-informed structural signals influence LLM SRL predictions, showing improvements in structural stability; and (3) an English–French projection case study illustrating how improved structural consistency can support downstream cross-lingual SRL projection.

These contributions collectively reposition SRL as a structurally grounded, efficient, and representation-consistent framework, aligning explicit predicate-argument modeling with modern encoder architectures, emerging LLM paradigms, and downstream cross-lingual SRL projection.[1]

2 Related Work

SRL research spans two decades, evolving from syntax-driven models to neural span encoders, and more recently, LLMs. This evolution reflects broader shifts in NLP modeling paradigms while preserving the core objective of explicit predicate-argument representation. We review prior work along four dimensions: structured SRL foundations, BERT-based architectures and the AllenNLP framework, LLM-based SRL, and cross-lingual SRL transfer. These threads contextualize our emphasis on efficiency, structural consistency, and explanatory adequacy within a dependency-informed, multilingual framework.

2.1 Structured SRL Foundations

Early SRL models emphasize syntactic and lexical cues. Through a series of CoNLL shared tasks (Carreras and Màrquez, 2005; Hajič et al., 2009; Pradhan et al., 2012) , these approaches evolve into neural architectures that incorporate dependency-path features (FitzGerald et al., 2015; Roth and Lapata, 2016; He et al., 2018). Most systems adopt PropBank conventions, assigning ARG0 to the agent, ARG1 to the theme, and ARG2 to the recipient. Over time, feature engineering with contextual embeddings has gradually replaced these approaches. An example of this is ELMo (Peters et al., 2018) and strong pre-trained baselines like GloVe (Stanovsky et al., 2018). Building on contextualized embeddings, Strubell et al. (2018) incorporate syntax-informed self-attention into SRL, encoding each sentence once and leveraging dependency signals across attention heads to achieve strong performance on CoNLL-2005 and CoNLL-2012.

2.2 BERT-based and AllenNLP SRL

Transformer-based architectures such as BERT (Devlin et al., 2019) have allowed for span-level BIO tagging as in Conia et al. (2022). BERT-based models (Shi and Lin, 2019; Fei et al., 2021) perform strongly via contextual embeddings and span-level decoding. Marcheggiani and Titov (2020) similarly incorporate constituency tree structures into BERT-based span representations via graph convolutional networks, achieving strong results on CoNLL-2005 and CoNLL-2012. AllenNLP (Gardner et al., 2018) generalizes this design for broad community use, integrating SRL, NER, coreference, and parsing.

Building on these advances, Shi and Lin (2019) implement a BERT-LSTM model that captures predicates using the [SEP] token, becoming the basis of AllenNLP's SRL implementation. However, post-2022 dependency and package incompatibilities have limited continued integration with modern frameworks such as spaCy (Honnibal and Montani, 2025) and PyTorch (Paszke et al., 2019). Our work revisits this encoder-based modeling framework, eliminating redundant per-predicate instance generation, improving inference efficiency, and introducing dependency-informed diagnostics for systematic structural analysis.

[1] Our Modernized SRL is at: https://anonymous.4open.science/r/modernized_SRL-CF96

More recently, Wang et al. (2022) implement a machine reading comprehension (MRC) encoder framework for SRL that jointly handles predicate identification and argument labeling, outperforming Shi and Lin (2019) on CoNLL-2012.

2.3 SRL in the LLM Era

Although LLMs excel at sequence modeling, explicit predicate–argument structure remains indispensable for faithfulness, controllability, cross-lingual transfer, and interpretability. SRL makes semantic relations explicit—this is challenging for LLMs despite their fluency in text generation.

Recent studies show that achieving competitive SRL performance with LLMs requires substantial structural scaffolding, including retrieval mechanisms, frame descriptions, and iterative correction strategies (Li et al., 2025). These findings suggest that structured semantic representations do not reliably emerge from generative inference alone. Prior studies indicate that LLMs struggle to assign consistent semantic roles across domains and languages without an auxiliary mechanism (Wu et al., 2022), and even advanced models (e.g., GPT-4) attain only partial accuracy as PropBank annotators (Bonn et al., 2024). Thus, structured role inventories such as PropBank (Jindal et al., 2022) and Universal PropBank (UP) (Pradhan et al., 2022) remain indispensable intermediate representations for reasoning and semantic transfer.

The recent survey by Chen et al. (2025) notes renewed interest in structured semantics as a complement to foundation models. Structured formulations of SRL—such as span-based parsing (Zhang et al., 2022) or dependency parsing (Shi et al., 2020)—make predicate-argument boundaries explicit. Even generative end-to-end approaches that match discriminative encoder performance on CoNLL-2012 rely on explicit role inventories and structured output constraints, underscoring that the predicate-argument structure remains central regardless of the modeling paradigm (Blloshmi et al., 2021). SRL thus bridges symbolic semantics and neural contextualization, providing interpretable scaffolding for faithful semantic reasoning.

Building on this perspective, we revisit structured SRL modeling, introducing a modernized framework that achieves comparable predictive performance with 10× faster inference using BERT, while RoBERTa and DeBERTa provide further gains. The framework integrates dependency-informed diagnostics for systematic structural analysis and representation-level evaluation of LLM behavior under explicit structural signals.

2.4 Cross-lingual SRL Transfer

Cross-lingual SRL transfer is now a central research focus, as annotated data remain scarce for most languages. However, transfer often propagates annotation errors and structural inconsistencies from English data to the target language. Our approach addresses this problem by detecting structural errors in English source sentences using a dependency-aware error detection method, improving the structural consistency of cross-lingual SRL projection in downstream settings.

Prior approaches such as divergence-aware SRL projection (Youm et al., 2024) and zero-shot conversational SRL (Wu et al., 2022) highlight the persistent challenge of maintaining consistent argument structures across typologically diverse corpora. Conia et al. (2021) address this through a unified cross-lingual SRL framework that operates across languages with heterogeneous annotation schemes without relying on word alignment, broadening the scope of cross-lingual transfer beyond projection-based approaches. Resources such as Universal PropBank 2.0 and PropBank's latest multilingual frames (Pradhan et al., 2022) now facilitate unified role mappings. These resources support and align with our model's dependency-aware design for cross-lingual expansion.

3 Structured SRL Framework

We fine-tune and evaluate our framework on CoNLL-2012. This section describes the dataset and the processing pipeline.

3.1 Data

We adopt the English portion of the CoNLL-2012 Shared Task dataset (Pradhan et al., 2012) derived from OntoNotes 5.0 (Weischedel et al., 2013),[2] using the official train, development, and test splits for benchmark consistency. The dataset provides span-based semantic role annotations grounded in PropBank, enabling systematic investigation of predicate-argument boundary prediction and structural consistency.[3]

We select CoNLL-2012 due to its span-level formulation of semantic roles, which aligns with our framework's emphasis on structural adequacy,

[2]We obtain this dataset from Linguistic Data Consortium (LDC), and all data processing follows LDC data use policies.

[3]Dataset creation details are provided in Appendix A.

argument-boundary stability, and dependency-informed diagnostics. This contrasts with token-level formulations such as CoNLL-2009, which target a different granularity of SRL.

3.2 Model

To construct the encoder-based structured SRL model, we condition the input on the predicate by appending a dedicated predicate span to the sentence representation:

`[CLS]|sent|[SEP]|pred|[SEP]`

Here `sent` and `pred` denote the sentence and predicate tokens, respectively. We follow the AllenNLP training paradigm to ensure comparability, but diverge at inference time by eliminating redundant per-predicate sentence encoding. AllenNLP constructs a separate instance for each predicate, resulting in repeated sentence tokenization and encoder recomputation. In contrast, our framework performs a single sentence-level tokenization and encoder pass and reuses them across all predicates in the sentence.

For comparison with our approach, we present Algorithm 1 to illustrate the code underlying AllenNLP's per-predicate encoding system.[4]

Algorithm 1 Original AllenNLP: Per-Predicate Encoding

```
function TextToInstance(tokens, labels)
    sent ← HFEncode(tokens)
    pred ← HFEncode(Select(tokens, labels=1))
    ids ← [CLS]|sent|[SEP]|pred|[SEP]
    tt ← [0]^{1+|sent|+1} || [1]^{|pred|+1};   am ← [1]^{|ids|}
    return Instance(MakeFields(ids, tt, am, labels, tokens))
function PredictSRL(model, tokenizer, inputs, p_idxs)
    INSTS ← []; P ← []
    for p ∈ p_idxs do
        labels ← OneHot(|inputs|, p)
        inst ← TextToInstance(inputs, labels)
        Push(INSTS, inst); Push(P, v_idxs)
    OUT ← ForwardInstances(model, INSTS)
    return PackBySentence(OUT, P)
```

Algorithm 2 presents our modernized inference design, which decouples sentence encoding from predicate-conditioned prediction. Our framework performs a single sentence-level encoding pass, ensuring consistent token indices across all predicate-conditioned structures within a sentence. The model reuses cached sentence representation to construct predicate-conditioned inputs. Importantly, each predicate still receives a predicate-conditioned contextualized encoder representation, preserving predicate-specific modeling capacity while eliminating redundant preprocessing. [5]

[4] AllenNLP's per-predicate encoding approach is openly available: https://docs.allennlp.org/models/main/models/structured_prediction/predictors/srl/.

[5] Full code details are provided in Appendix B.

We retain the first wordpiece index for each original token and use these indices to map predicate-conditioned predictions back to word-level BIO labels, ensuring compatibility with CoNLL-style span annotations.

Algorithm 2 Our Approach: Modernized SRL

```
function EncodeSentenceOnce(inputs, tokenizer)
    wp ← tokenizer(inputs, split = T, specials = F)
    sent ← Flatten(wp.ids)
    f_wp_idxs ← FirstPosPerWord(wp.word_ids(), +1)
    return (sent, f_wp_idxs)
function PredictSRL(model, tokenizer, inputs, p_idxs)
    (sent, f_wp_idxs) ← EncodeSentenceOnce
                        (inputs, tokenizer)
    for p ∈ p_idxs do
        IDS, TT, WF, AM, P ← []
        pred ← Flatten(tokenizer(inputs[p]))
        ids ← [CLS]|sent|[SEP]|pred|[SEP]
        tt ← [0]^{1+|sent|+1} || [1]^{|pred|+1}, am ← [1]^{|ids|}
        Push(IDS, ids); Push(TT, tt); Push(AM, am);
        Push(P, p); Push(WF, f_wp_idx)
    INPUTS ← PadAndStack(IDS, TT, AM)
    OUT ← Forward(model, INPUTS, P, WF)
    return PackBySentence(OUT, P)
```

Although sentence-level representations are shared, predicate-specific distinctions are introduced via the predicate span and segment embeddings, ensuring that argument predictions remain predicate-conditioned rather than sentence-global. We then evaluated the predictive performance and efficiency of our model on benchmark data.

4 Experiments and Results

The goal of our framework is efficient structured inference: comparable accuracy to strong baselines, with substantially faster inference to support downstream tasks.

We follow the AllenNLP SRL configuration based on the SRL-BERT model by Shi and Lin (2019). The LSTM and MLP hidden dimensions are 768 and 300, respectively, with a predicate indicator embedding size of 10. Using a BERT-cased encoder and a learning rate of 5×10^{-5}, we conduct all experiments on AMD EPYC 9655P (3 CPUs) and NVIDIA DGX B200 machines (1 GPU). We use BERT-based models as the primary comparison model against AllenNLP because BERT represents the community-standard baseline transformer encoder for SRL and offers the most efficient inference speed among the evaluated encoders. RoBERTa and DeBERTa are included to assess performance sensitivity to backbone choice.

On the test set (11,289 sentences), our model runs 10× faster than the AllenNLP baseline (1.32 vs. 13.18 min) while maintaining nearly identical accuracy. The dataset contains 32,617 clause-level

predictions across 877,873 tokens. At the phrase level, our model achieves comparable performance (F1: 86.15 vs. 86.72), with Precision/Recall scores of 85.98/86.32 versus 86.81/86.63 for AllenNLP. Table 1 summarizes these results.

Metric	Our Model (BERT)	Our Model (RoBERTa)	Our Model (DeBERTa)	AllenNLP
P (%)	85.98	86.19	**86.78**	86.81
R (%)	86.32	87.97	**88.22**	86.63
F1 (%)	86.15	87.07	**87.49**	86.72
Time (m)	**1.36**	5.36	7.57	13.18

Table 1: Comparison of our modernized BERT-based SRL model and AllenNLP on the OntoNotes 5.0 test set, showing comparable accuracy with a 10× speedup.

Partition	Subcase	% within partition	% of total
Agreement (95.20%)	Both correct	88.49%	84.24%
	Both wrong	11.51%	10.96%
Disagreement (4.80%)	Allen correct	43.96%	2.11%
	Ours correct	29.29%	1.41%
	Neither correct	26.75%	1.28%

Table 2: We divide prediction outcomes relative to the ground truth into *agreement* (835,699 tokens), where both systems produce identical predictions, and *disagreement* (42,174 tokens), where their outputs diverge. Within the agreement set (95.2% of the total), both systems are correct for 734,415 tokens (88.49% of the within agreement). Within the disagreement set, AllenNLP is correct for 18,539 tokens (43.96%), and our system for 12.352 tokens (29.29%).

Across encoder variants, predictions exhibit high agreement with AllenNLP outputs (BERT: 95.2%, RoBERTa: 97.97%, DeBERTa: 94.99%) For broader context, Blloshmi et al. (2021) achieves 87.3 F1 on CoNLL-2012, indicating that our DeBERTa result of 87.49% is competitive with strong prior systems while remaining significantly faster.

Of the 877,873 tokens, 841,002 tokens (95.2%) match AllenNLP outputs on the same unseen test set, confirming replication fidelity. The remaining 4.8% (42,174 tokens) represent token-level mismatches analyzed next.

Table 2 presents a breakdown of the distribution of these predictions relative to the OntoNotes 5.0 ground truth, distinguishing between agreement and disagreement cases, as well as whether each system is correct. Among agreement cases—where both systems produce identical outputs—88.49% are correct (accounting for 84.24% of the total), and 11.51% are incorrect (10.96% of the total). In the disagreement cases, 43.96% of AllenNLP's predictions are correct (2.11% of the total), and 29.29% of our system are correct (1.41% of the total). The remaining 26.75% (1.28% of the total) are cases where neither system is correct.

5 Analysis and Remediation

Although overall performance is strong, understanding residual errors is essential. Modern encoder models rely on self-attention mechanisms, calculating representations based on weighted interactions between token vectors. Because encoder representations are learned from token interactions rather than explicit span constraints, they can produce structurally inconsistent BIO sequences. To analyze mismatches, we employ a *dependency-aware* analyzer that automatically merges repeated spans when possible and flags the rest for human review. This process resolves a measurable subset of mismatches and systematically categorizes the rest. Fine-grained character-, token-, and span-level comparisons make manual review impractical. Our analyzer distinguishes between automatic and human-resolvable errors, thereby enhancing the structural interpretability of model behavior.

Before detailing the dependency-aware error detection framework, we first examine which semantic role categories are most frequently missed in the 5% of predictions where our system and AllenNLP diverge. This analysis lays the foundation for characterizing the sources of residual errors.

5.1 Analysis of Missing Roles in Ours and AllenNLP

We investigate missed arguments via one-to-one comparisons between our system and AllenNLP outputs, evaluated against the OntoNotes 5.0 ground truth. Relative to our system, `ARGM` accounts for 48.05% of differences across argument categories. Core roles `ARG0`-`ARG2` together account for 48.77% of mismatches, with `ARG1` and `ARG2` the most prominent. Among `ARGMs`, `ARGM-ADV` (adverbial modifier) represents the largest share of discrepancies (28.33%), followed by `ARGM-MNR` (11.97%) and `ARGM-TMP`, with `ARGM-LOC` forming the next substantial portion.[6]

We present role-wise differences between AllenNLP predictions and the ground truth. The largest share of missing labels is `ARGM` (48.10%). The core roles `ARG0`, `ARG1`, and `ARG2` together represent 48.42% of the differences. Within `ARGMs`, a small subset of adjunct types dominates missing labels: `ARGM-ADV` (25.72%), `ARGM-PRD`

[6]Full diagnostic distributions are provided in Appendix C.

(13.13%), followed by ARGM-MNR, ARGM-TMP, and ARM-LOC—accounting for over 75% of ARGM-related omissions.[7]

While this analysis identifies dominant omission patterns, it does not explain the structural causes. Therefore, the next section introduces a dependency-aware diagnostic framework to examine span-level sources of inconsistency.

5.2 Dependency-Aware Error Analysis

Our model predicts span-based semantic roles, and some errors arise from BIO tagging mistakes that result in span violations. In particular, we detect cases where the beginning of an argument is incorrectly assigned multiple times to the same semantic role, as illustrated here for B-ARG0. Our analyzer identifies and corrects errors by enforcing proper span boundaries using dependency information. The opening bracket ("[") marks the start of a span, and all subsequent tokens within that span receive I-tags until the closing bracket ("]").

```
[B-ARG0: temperament] [B-ARG0: through
I-ARG0: study I-ARG0: and I-ARG0:
practice]
```

Among 4.8% of divergent predictions between AllenNLP and our system, span violations account for 3.8% of all spans in all systems' predictions (17,387 spans) and 3.79% of all spans in the AllenNLP outputs (19,131 spans). In both systems, ARGM-ADJ exhibits the highest number of span violations, representing 16.67% of such cases in our model and 10.00% in AllenNLP.

Because PropBank-style annotation prohibits duplicate argument roles per predicate (e.g., two ARG0 spans for the same verb) (Jurafsky and Martin, 2025), repeated spans for the same semantic role are structurally invalid. This constraint—long established in linguistic theory—requires that each argument occur only once per predicate. Thus, when our system produces multiple spans for the same role, it merges them into a single continuous argument span to satisfy this constraint.

In such cases, the first token carries the *B* tag, and all subsequent tokens carry the *I* tag, rather than restarting with *B*, as illustrated below:

```
[B-ARG0: temperament I-ARG0: through
I-ARG0: study I-ARG0: and I-ARG0:
practice]
```

Our error analyzer operates over BIO-tagged token sequences and dependency parsers.[8] It identifies adjacent spans assigned the same semantic role as candidate repeated arguments, whose dependency structure determines classification. Spans attached to the same syntactic head are labeled as same_head; cases involving prepositional modification are categorized as pp_attach; and spans lying within another's dependency subtree are classified as subtree_attach (see Table 3).

We mark errors classified into predefined types as "Fixable"; otherwise, sentences are flagged with "Review Required", indicating that the error cannot be automatically resolved. Classification outcomes enable structured evaluation of error patterns across systems. We apply dependency-aware error detection to cases where AllenNLP predicts correct tags, but our model does not—accounting for 2.11% of tokens within the 4.8% total disagreement. Across 18,539 such errors, our analyzer identifies 498 tokens (2.68%) as belonging to fixable categories (same_head, subtree_attach, and pp_attach) that can be automatically resolved. Additionally, 933 tokens (5.03%) involve repeated role spans that cannot be resolved automatically and are therefore flagged as "Review Required" (see Table 4).

The dependency-aware analyzer provides a systematic framework for characterizing residual inconsistencies. Leveraging structural signals, it identifies potential mislabeling, flags cases needing human judgment, and improves the structural interpretability of SRL predictions.

6 Representation Behavior of LLMs

Although LLMs are extremely capable at text generation, adapting them for SRL tasks introduces tradeoffs of accuracy and efficiency. Without a heavy computational stack, LLMs are often slower and less accurate than the proposed encoder framework. To understand this premise, we investigate how LLMs perform SRL when provided with explicit syntactic structure. We examine whether externally supplied dependency information influences SRL predictions and how reliably LLMs utilize such signals. To enable controlled analysis, we compare LLM outputs with and without dependency-informed context, isolating the interaction between syntactic structure and generative semantic inference. We conduct experiments in a zero-shot setting using gpt-oss-120b.

To probe LLM SRL behavior under controlled conditions, we adopt an instructional guidance

[7] Full diagnostic distributions are provided in Appendix D.
[8] See Appendix E for the analyzer pipeline.

Subtype	Definition	Example
same_head	Both spans attach to the same syntactic head.	✓ *Ours: [ARG2: just brain dead]* ✗ *Allen: [ARG2: just] [ARG2: brain dead]* ✗ *Ours: [ARG1: the various revisions] to [ARG1: trade law]* ✓ *Allen: [ARG1: the various revisions to trade law]*
pp_attach	Right (left) span is a PP modifying the left (right) span.	✓ *Ours: [ARG1: a long section of road]* ✗ *Allen: [ARG1: a long section] [ARG1: of road]* ✗ *Ours: [ARG1: several kinds] of [ARG1: microbes]* ✓ *Allen: [ARG1: several kinds of microbes]*
subtree_attach	One span lies inside the dependency subtree of the other span's head.	✓ *Ours: [ARGM-TMP: for the fourth time on Friday]* ✗ *Allen: [ARGM-TMP: for the fourth time] [ARGM-TMP: on Friday]* ✗ *Ours: [ARG0: anyone who] finds [ARG0: the pair]* ✓ *Allen: [ARG0: anyone who finds the pair]*

Table 3: Error subtypes, definitions, and illustrative examples. All types reflect **repeated same arguments** within a sentence. These examples are drawn from 4.8% of cases where predictions disagree between AllenNLP and our system. In this example, we present a case in which one of the two systems produces the correct prediction.

Error bucket	Tokens	% of 18,539
NO_BUCKET	17,108	92.28
OTHER_REPEAT	933	5.03
same_head	314	1.67
subtree_attach	141	0.76
pp_attach	43	0.23

Table 4: Token distribution for cases where AllenNLP is correct on disagreements with our model (18,539 instances). Errors in the five structural categories (`same_head`, `subtree_attach`, and `pp_attach`) are labeled "Fixable", while others are flagged as "Review Required," indicating a need for human intervention.

formulation strategy,[9] providing explicit role definitions and structural output constraints on the CoNLL-2012 test set. We compare predictions with and without dependency-informed syntactic signals, enabling a controlled analysis of how external structural signals influence SRL performance.

Table 5 shows that dependency-informed prompts improve predictions (F1: 42.17 vs. 44.30). The gains reflect improved span-boundary stability and phrase-level coherence rather than role classification accuracy, indicating more reliable argument grouping. Unlike prior LLM-based SRL approaches that require retrieval or iterative self-correction (Li et al., 2025), our formulation remains lightweight, relying on structural guidance and dependency cues.

Qualitative analysis illustrates these structural effects. For the predicate *fetches* in *Fruit that shows a hint of black through the red color of the skin fetches the best price*, LLMs without dependency cues collapse the ARG0 span to just *Fruit*. With dependency information, the model preserves the full phrase (*Fruit that shows a hint of black through the red color of the skin*), indicating an improvement in structural grouping.

The second pattern involves duplicate role assignments. In *to publicly support the project*, predictions without dependency signals produce fragmented ARG1 spans *[B-ARG1: to I-ARG1: publicly] [B-ARG1: support I-ARG1: the I-ARG1: project]*. Dependency-informed prompts instead yield a single continuous span, reflecting constraint-consistent phrase composition.

These findings indicate that dependency-informed structural signals enhance representation stability rather than classification capacity, highlighting the interaction between syntactic structure and generative semantic inference.

7 Multilingual Applicability: A Projection Case Study

Building on our English benchmark results, we examine downstream multilingual applicability through an English–French projection case study. Cross-lingual SRL remains challenging across language pairs and annotation schemes (Conia et al., 2021); our framework addresses a key source of this difficulty by improving structural accuracy in the English source before projection. We adopt cross-lingual SRL projection techniques to examine how explicitly represented predicate-argument structures transfer across languages, aligning English–French sentence pairs via one-to-one token mappings prior to semantic role transfer.

Recent work on AllenNLP-based cross-lingual SRL transfer highlights hallucinations from structural mismatches between languages (Youm et al.,

[9] See Appendix F for the full prompt.

Method	Struct. Const.	RAG	Self-Corr.	Precision	Recall	F1
Li et al. (2025) (LLaMA-8B)	✗	✓	✓	6.29	7.59	6.88
Ours (gpt-oss-120b)	✓	✗	✗	46.58	35.58	42.17
Ours (gpt-oss-120b with Dependency-Guided Structure)	✓	✗	✗	**49.03**	**40.44**	**44.30**

Table 5: Comparison of SRL performance across inference strategies on the CoNLL 2012 test set. Struct. Const. (Structural Constraints) denotes schema-constrained prompts enforcing BIO tagging rules. All evaluations are conducted in a zero-shot setting without model fine-tuning.

2024). The approach aligns English-French sentence pairs by adding or removing tokens for one-to-one mappings before semantic role transfer, but still carries over errors from English SRL outputs.

Figure 2 illustrates such a case. When aligning *after the October 1987 crash* with its French counterpart *après l'écrasement (de) octobre 1987*, *de* is omitted and *écrasement* is repeated. However, on the English side, *after* is incorrectly tagged as `I-ARGM-TMP` rather than `B-ARGM-TMP`, leaving the correction incomplete and propagating the error to the French projection.

[I-ARGM-TMP] after	—	après; 7-5
[I-ARGM-TMP] the	—	l' ; 8-6
[I-ARGM-TMP] october	—	écrasement ; 9-7
[I-ARGM-TMP] october	—	octobre ; 9-9
[I-ARGM-TMP] 1987	—	1987 ; 10-10
[I-ARGM-TMP] crash	—	écrasement ; 11-7

Figure 2: Prior cross-lingual SRL (Youm et al., 2024) resolves alignment errors before SRL transfer. The English excerpt (tokens 7-11) aligns with the French excerpt (tokens 5-10). English tokens *october* (9) and *crash* (11) align to French *écrasement* (7). The spurious 9-7 alignment is ultimately excluded, but *after*—and correspondingly *après*—are left incorrectly tagged as `I-ARG-TMP`.

We address this limitation by replacing the AllenNLP SRL component with our structure inference framework, preserving predicate-argument boundaries while eliminating redundant per-predicate processing. This restructuring yields $\approx 10\times$ faster projection and makes large-scale dataset construction more feasible.

Beyond efficiency, our model improves structural accuracy. In the same example, the dependency-aware detector recognizes *after the October 1987 crash* as a single noun phrase and labels it under `pp_attach`. It then correctly initiates the span with a `B-ARGM-TMP` tag on *after*, rather than the erroneous `I-ARGM-TMP` label in Figure 2.

[B-ARGM-TMP] after	—	[B-ARGM-TMP] après
[I-ARGM-TMP] the	—	[I-ARGM-TMP] l'
[I-ARGM-TMP] october	—	[I-ARGM-TMP] octobre
[I-ARGM-TMP] 1987	—	[I-ARGM-TMP] 1987
[I-ARGM-TMP] crash	—	[I-ARGM-TMP] écrasement

Figure 3: Dependency-aware correction applied to the *October 1987 crash* example. The revised SRL output correctly marks after with a `B-ARGM-TMP` tag, restoring the missing boundary and producing a complete span alignment across English and French.

These findings underscore a broader implication: structurally explicit semantic representations provide stability advantages not only within monolingual inference but also during cross-lingual projection, where small boundary deviations can amplify into large inconsistencies. Leveraging this structure-preserving design, our framework supports structurally faithful SRL projection in downstream multilingual settings. While illustrated here through an English-French case study, this analysis highlights the potential value of structured SRL for multilingual applications that leverage English-centric resources.

8 Conclusion and Future Work

This paper revisits structured semantic role labeling (SRL) to demonstrate the continued value of explicit predicate-argument representations in modern NLP. We introduce a modernized encoder-based framework that eliminates redundant per-predicate encoding while preserving predictive performance. By caching sentence-level representations, the framework achieves a $10\times$ inference speedup over AllenNLP while maintaining comparable accuracy and improving performance with encoders such as RoBERTa and DeBERTa. This efficiency is especially useful for downstream settings requiring repeated predicate inference.

We also introduce a dependency-informed diagnostic methodology for systematically characterizing span boundary inconsistencies. The analysis shows that dependency-informed structural signals primarily enhance representational stability in both encoder-based and LLM-based SRL, improving span coherence and argument grouping rather than raw classification accuracy. The framework aligns naturally with cross-lingual SRL projection pipelines, with implications for downstream multilingual analysis.

Future work includes fine-tuning decoder-only LLMs within this structured formulation, integrating SRL transfer pipelines for multilingual dataset construction, and refining dependency-aware error detection to further reduce regression cases.

Limitations

We analyze the errors produced by our SRL models and categorize them using a dependency-aware detector. While the detector identifies systematic split-span patterns, the majority of cases still require human judgment for comprehensive resolution. To mitigate this, we report which errors are automatically solvable according to our rules and which are not safely fixable without review. This yields an actionable map of error types, clarifies the limits of automatic repair, and guides future SRL diagnostics by focusing human effort where it is most impactful.

Ethical Considerations

Our SRL model relies on transformer backbones trained on OntoNotes, which may encode the domain, genre, and annotators' narrow view present in that corpus. As a result, the model can systematically underperform—or "penalize"—certain argument types or constructions that are underrepresented in OntoNotes, and these errors are not always fixable by automatic post-hoc analysis. To mitigate this, we encourage training and evaluation on broader, contemporary corpora, and we release rules/code to enable independent audits. Through this process, future work aims to narrow gaps rather than overfitting to OntoNotes.

References

Rexhina Blloshmi, Simone Conia, Edoardo Barba, Tommaso Pasini, 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 (IJCAI-21)*, pages 3786–3793.

Claire Bonial, Julia Bonn, Kathryn Conger, Jena D. Hwang, and Martha Palmer. 2014. PropBank: Semantics of new predicate types. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3013–3019, Reykjavik, Iceland. European Language Resources Association (ELRA).

Julia Bonn, Harish Tayyar Madabushi, Jena D. Hwang, and Claire Bonial. 2024. Adjudicating LLMs as PropBank adjudicators. In *Proceedings of the Fifth International Workshop on Designing Meaning Representations @ LREC-COLING 2024*, pages 112–123, Torino, Italia. ELRA and ICCL.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

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*.

Simone Conia, Andrea Bacciu, and Roberto Navigli. 2021. Unifying cross-lingual semantic role labeling with heterogeneous linguistic resources. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 338–351, Online. Association for Computational Linguistics.

Simone Conia, Edoardo Barba, Alessandro Scirè, and Roberto Navigli. 2022. Semantic role labeling meets definition modeling: Using natural language to describe predicate-argument structures. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jing Fan, Dennis Aumiller, and Michael Gertz. 2023. Evaluating factual consistency of texts with semantic role labeling. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 89–100, Toronto, Canada. Association for Computational Linguistics.

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*, volume 35, pages 12803–12811.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 960–970, Lisbon, Portugal. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, M. 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: Shared Task*, pages 1–18, Boulder, Colorado. Association for Computational Linguistics.

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, Vancouver, Canada. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Shexia He, Zuchao Li, Hai Zhao, and Hongxiao Bai. 2018. Syntax for semantic role labeling, to be, or not to be. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2061–2071, Melbourne, Australia. Association for Computational Linguistics.

Matthew Honnibal and Ines Montani. 2025. spaCy: Industrial-strength natural language processing in python.

Ishan Jindal, Alexandre Rademaker, Michał Ulewicz, Linh Ha, Huyen Nguyen, Khoi-Nguyen Tran, Huaiyu Zhu, and Yunyao Li. 2022. Universal proposition bank 2.0. In *Proceedings of the 13th Language Resources and Evaluation Conference (LREC)*, pages 1700–1711, Marseille, France. European Language Resources Association.

Daniel Jurafsky and James H. Martin. 2025. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*, 3rd edition.

Manfred Klenner and Anne Göhring. 2022. Semantic role labeling for sentiment inference: A case study. In *Proceedings of the 18th Conference on Natural Language Processing (KONVENS 2022)*, pages 144–149, Potsdam, Germany. KONVENS 2022 Organizers.

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, Vienna, Austria. Association for Computational Linguistics.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. AAAI'19/IAAI'19/EAAI'19. AAAI Press.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Diego Marcheggiani and Ivan Titov. 2020. Graph convolutions over constituent trees for syntax-aware semantic role labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3915–3928, Online. Association for Computational Linguistics.

Brodie Mather, Bonnie Dorr, Adam Dalton, William de Beaumont, Owen Rambow, and Sonja Schmer-Galunder. 2022. From stance to concern: Adaptation of propositional analysis to new tasks and domains. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3354–3367, Dublin, Ireland. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, and 2 others. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sameer Pradhan, Julia Bonn, Skatje Myers, Kathryn Conger, Tim O'Gorman, James Gung, and Martha Palmer. 2022. Propbank comes of age—larger, smarter, and more diverse. In *Proceedings of *SEM (STARSEM) 2022*, pages 278–288.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Proceedings of the Joint Conference on EMNLP and CoNLL – Shared Task*, pages 1–40, Jeju, Korea. Association for Computational Linguistics.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume*

1: Long Papers), pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

Tianze Shi, Igor Malioutov, and Ozan Irsoy. 2020. Semantic role labeling as syntactic dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7551–7571, Online. Association for Computational Linguistics.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895, New Orleans, Louisiana. Association for Computational Linguistics.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

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 (COLING 2022)*, pages 2193–2205, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Peter Kaufmann, Michelle Franchini, Mohammed El-Bachouti, Robert Houston, and Olga Krasny. 2013. Ontonotes release 5.0. LDC2013T19.

Han Wu, Haochen Tan, Kun Xu, Shuqi Liu, Lianwei Wu, and Linqi Song. 2022. Zero-shot cross-lingual conversational semantic role labeling. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 269–281.

Sangpil Youm, Brodie Mather, Chathuri Jayaweera, Juliana Prada, and Bonnie Dorr. 2024. Dahrs: Divergence-aware hallucination-remediated srl projection. In *International Conference on Applications of Natural Language to Information Systems*, pages 423–438. Springer.

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 (COLING)*, pages 4212–4227.

```
{
'words': ['Mary', 'gave', 'me', 'a', 'present',
'.'],
'predicate_word_idx': 1,
'labels': ['B-ARG0', 'B-V', 'B-ARG2', 'B-ARG1',
'I-ARG1', 'O']
}
```

Figure 4: Example SRL instance represented as a JSON object, showing tokenized words, the predicate index, and BIO-formatted semantic role labels.

A Dataset Creation

In CoNLL-2012, each predicate in a sentence yields a separate SRL instance, resulting in multiple entries for sentences containing more than one predicate. As shown in Figure 4, each instance is stored as a JSON object containing:

- `words`: the tokenized sentence;
- `predicate_word_idx`: the index of the predicate token; and
- BIO-formatted sequence of semantic role tags (e.g., `["B-ARG0", "B-V", "B-ARG2", "B-ARG1", "I-ARG1", "O"]`).

To mirror AllenNLP's data format, we reconstruct tokenized `.gold_skel` files from OntoNotes `.onf` files as follows:

- **Token extraction:** Extract tokens and their indices from each parse tree, removing artifacts such as "$*$" or "%."
- **Gold skeleton alignment:** Align the extracted tokens with corresponding `.gold_skel` sections, replacing `[WORD]` placeholders with surface tokens.
- **Semantic role reconstruction:** Convert column-based SRL annotations (columns ≥ 12 in CoNLL format) into BIO tags using a custom parser that handles nested and single-token spans.
- **JSON object generation:** For each predicate, build a JSON object capturing the predicate index, argument role spans (`ARG0`, `ARG1`, `ARG2`, etc.), and the complete BIO label sequence. Sentences with misaligned tokens are logged and excluded.

This preprocessing yields a large, sentence-aligned set of predicate-level instances for fine-tuning and evaluation, fully compatible with prior SRL systems. Each example represents a specific predicate and its arguments.

B Algorithm Walkthrough

This section provides line-by-line descriptions of the two inference algorithms presented in Section 3.

Algorithm 1: Original AllenNLP Per-Predicate Encoding

- **Line 1:** Define `TextToInstance` — a helper function responsible for constructing a per-predicate instance for semantic role labeling (SRL).
- **Line 2–3:** Encode the input tokens into subword IDs (`sent`) using the Hugging Face tokenizer (`HFEncode`). Select and encode the predicate token (`pred`).
- **Lines 4–5:** Construct the model input sequence: concatenate predicate with the sentence, `[CLS]|sent|[SEP]|pred|[SEP]`; assign segment IDs (`tt`) to distinguish sentence and predicate parts (0 for `sent`, 1 for `pred`); create the attention mask (`am`) to indicate valid tokens.
- **Line 6:** Return a structured `INSTANCE` object containing the constructed fields (`ids`, `tt`, `am`, `labels`, and `tokens`) for downstream modeling, and end of the function.
- **Line 7:** Define `PredictSRL(model, tokenizer, inputs, p_idxs)` — the main SRL inference routine, which processes input sentences and predicate indices.
- **Line 8:** Initialize an empty list of per-predicate instances (`INSTS`) and a mapping of predicates to their source sentences (`P`).
- **Lines 9–12:** Iterate over all predicate indices. For each predicate position `p`: create a one-hot label vector (`labels`); build a per-predicate instance using `TextToInstance` function (line 1-7); append the instance to `INSTS`.
- **Line 13:** Perform a forward pass over the model with all instances (`INSTS`) to generate per-predicate predictions (`OUT`).
- **Line 14:** Regroup predictions by their original sentences using `PackBySentence(OUT, P)` and return the final output: a list of SRL-labeled, BIO-marked sentences.

Algorithm 2: Our Modernized SRL

- **Line 1:** Define `EncodeSentenceOnce`—a helper function that performs sentence-level encoding once, avoiding redundant recomputation across predicates.
- **Lines 2–5:** Perform sentence-level encoding and preprocessing: tokenize into wordpieces, flatten token IDs, and record first wordpiece indices. Then, return preprocessed sentence representation (`sent`) and wordpiece index mapping (`f_wp_idxs`).
- **Line 6:** Define main SRL inference function—(`PredictSRL(model, tokenizer, inputs, v_idxs)`.
- **Line 7:** Encode the sentence once using `EncodeSentenceOnce` (line 1-7) to avoid redundant sentence-level encoding and improve efficiency.
- **Lines 8–14:** Iterate over each predicate and build model inputs: initialize empty containers, `IDS`, `TT`, `WF`, `AM`, and `P`; tokenize the predicate span; and construct full input by concatenating the reused sentence token IDs with predicate token ID, `[CLS]|sent|[SEP]|pred|[SEP]`; build segment IDs (0 for `sent`, 1 for `pred`); create the attention mask (`am`) to indicate valid tokens.
- **Line 15:** Combine the per-predicate input tensors into a batch with `PadAndStack(IDS, TT, AM)` for model inference.
- **Line 16:** Perform the forward pass: `OUT` ← `Forward(model, INPUTS, P, WF)`, producing contextualized token representations that are mapped to predicate-specific logits.
- **Line 17:** Regroup predictions by their original sentences using `PackBySentence(OUT, P)` and return the final output: a list of SRL-labeled, BIO-marked sentences.

C Distribution of Missing Semantic Roles—Our system

This section summarizes the distribution of missing roles in **our system** relative to the ground truth. Figure 5 shows the proportions of roles missing from our predictions, while Figure 6 presents the proportions of missing modifiers (ARGMs) from our prediction.

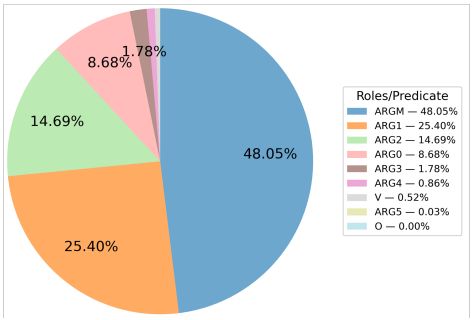

Figure 5: Proportion of missing roles or predicates in **our system's** predictions compared to OntoNotes 5.0 ground truth.

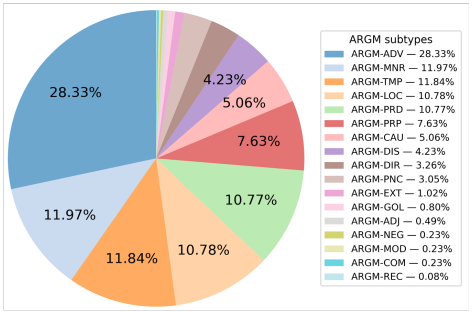

Figure 6: Proportion of missing ARGMs in **our system's** predictions compared to OntoNotes 5.0 ground truth.

D Distribution of Missing Semantic Roles—AllenNLP

This section summarizes the distribution of missing roles in **AllenNLP** relative to the ground truth. Figure 7 reports the proportions of roles missing from AllenNLP, while Figure 8 shows the proportions of missing modifiers (ARGMs) from AllenNLP prediction.

E Error Analyzer Overview

Figure 9 presents the dependency-aware error analysis pipeline, detailing the sequential steps used to detect adjacent spans, assess semantic role consistency, and categorize structural inconsistencies.

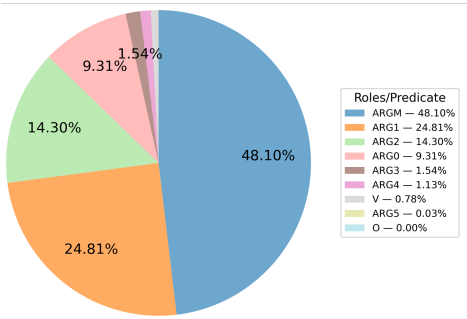

Figure 7: Proportion of missing roles or predicates in **AllenNLP's** predictions compared to OntoNotes 5.0 ground truth.

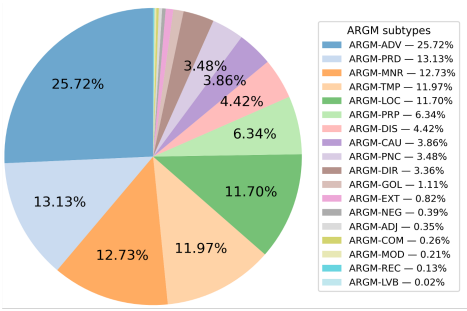

Figure 8: Proportion of missing ARGMs in **AllenNLP's** predictions compared to OntoNotes 5.0 ground truth.

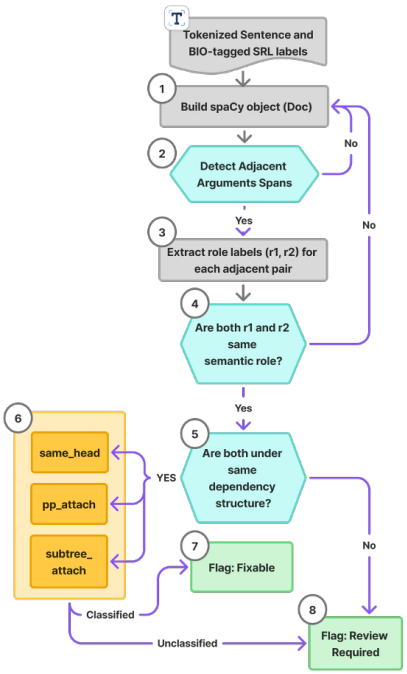

Figure 9: Overview of the modernized Dependency Aware Error Analyzer.

F Example Prompts for LLM Evaluation

This section presents the prompt used for dependency-informed SRL, shown both **with** and **without** dependency information in the LLM.

Prompt with dependency information for SRL

sent (MUST NOT change): I want to go home

Phrase hints (token spans over Words):
NP0: NP [0:1], head=0, text="I"
VP3: VP [2:5], head=3, text="to go home"
XCOMP1: XCOMP [2:5], head=3, text="to go home"
VP2: VP [0:5], head=1, text="I want to go home"

Protected phrase IDs (DO NOT split SRL arguments inside these spans): ["XCOMP1"]

Return SRL in EXACTLY this JSON schema (example):

```
{
  "sent": "",
  "words": ["..."],
  "verbs": [
    {"verb": "...", "label": ["O","B-ARG0","I-ARG0","B-V"]}
  ]
}
```

Hard rules:
(1) words is the list of tokens obtained from the sentence, separated by whitespace (2) For each verb, label length MUST equal len(words). (3) BIO: I-X cannot start a span; it must follow B-X/I-X of the same X. (4) Include ALL predicates. (5) Use phrase hints ONLY as boundary guidance; do NOT create new tokens. (6) Phrase integrity: do NOT split a role into multiple segments inside any protected phrase span. **Bad:** [ARG1: to] [ARG1: go home] when 'to go home' is protected.

Label guide:
Core: ARG0 agent; ARG1 patient; ARG2 instrument/benefactive/attribute; ARG3 start/benefactive/attribute; ARG4 end; ARG5 predicate-dependent.
Modifiers: ARGM-EXT extent; ARGM-LOC location; ARGM-DIR direction; ARGM-NEG negation; ARGM-MOD general-mod; ARGM-ADV adverbial; ARGM-MNR manner;
ARGM-PRD secondary-pred; ARGM-REC reciprocal; ARGM-TMP temporal; ARGM-PRP purpose; ARGM-PNC purpose-no-cause; ARGM-CAU cause; ARGM-ADJ adjectival (nouns);
ARGM-COM comitative; ARGM-DIS discourse; ARGM-DSP direct-speech; ARGM-GOL goal; ARGM-LVB light-verb (nouns); ARGA secondary-agent; ARGM-PRR predicating-relation.

Prompt without dependency information for SRL

sent (MUST NOT change): I want to go home

Return SRL in EXACTLY this JSON schema (example):

```
{
  "sent": "",
  "words": ["..."],
  "verbs": [
    {"verb": "...", "label": ["O","B-ARG0","I-ARG0","B-V"]}
  ]
}
```

Hard rules:
(1) words is the list of tokens obtained from the sentence, separated by whitespace (2) For each verb, label length MUST equal len(words). (3) BIO: I-X cannot start a span; it must follow B-X/I-X of the same X. (4) Include ALL predicates in the sentence.

Label guide:
Core: ARG0 agent; ARG1 patient; ARG2 instrument/benefactive/attribute; ARG3 start/benefactive/attribute; ARG4 end; ARG5 predicate-dependent.
Modifiers: ARGM-EXT extent; ARGM-LOC location; ARGM-DIR direction; ARGM-NEG negation; ARGM-MOD general-mod; ARGM-ADV adverbial; ARGM-MNR manner;
ARGM-PRD secondary-pred; ARGM-REC reciprocal; ARGM-TMP temporal; ARGM-PRP purpose; ARGM-PNC purpose-no-cause; ARGM-CAU cause; ARGM-ADJ adjectival (nouns);
ARGM-COM comitative; ARGM-DIS discourse; ARGM-DSP direct-speech; ARGM-GOL goal; ARGM-LVB light-verb (nouns); ARGA secondary-agent; ARGM-PRR predicating-relation.