Promptly Predicting Structures: The Return of Inference

Maitrey Mehta[♠], **Valentina Pyatkin**[♣,♢], **and Vivek Srikumar**[♠]

[♠]Kahlert School of Computing, University of Utah
[♢]University of Washington
[♣]Allen Institute for AI
{maitrey,svivek}@cs.utah.edu, valentinap@allenai.org

Abstract

Prompt-based methods have been used extensively across NLP to build zero- and few-shot label predictors. Many NLP tasks are naturally structured: that is, their outputs consist of *multiple* labels which constrain each other. Annotating data for such tasks can be cumbersome. *Can the promise of the prompt-based paradigm be extended to such structured outputs?* In this paper, we present a framework for constructing zero- and few-shot linguistic structure predictors. Our key insight is that we can use structural constraints—and combinatorial inference derived from them—to filter out inconsistent structures predicted by large language models. We instantiated this framework on two structured prediction tasks, and five datasets. Across all cases, our results show that enforcing consistency not only constructs structurally valid outputs, but also improves performance over the unconstrained variants.

1 Introduction

Structured prediction requires making multiple inter-dependent structurally constrained decisions (Smith, 2010; Nowozin et al., 2014). Many NLP tasks are naturally structured. Consider, for instance, the task of Semantic Role Labeling (SRL) that aims to identify the semantic roles played by sentence constituents for a predicate (Palmer et al., 2010). Given a sentence '*Elrond gave Aragorn the sword.*' and predicate *gave*, the task entails extracting its semantic arguments — the giver (`Agent`), the receiver (`Recipient`), and the thing given (`Theme`). These decisions are mutually constrained; e.g., the arguments have to be distinct phrases.

Existing methods for structured prediction rely on large labeled datasets. However, obtaining structured labels is not straightforward, especially from non-experts. It invariably requires detailed annotation guidelines about the task, the label set, and the interactions between labels. Methods like the re-formulation of structured prediction tasks into a question-answering (QA) format (He et al., 2015; Levy et al., 2017; Du and Cardie, 2020; Pyatkin et al., 2020, 2023) intend to make such tasks more amenable for annotation. However, procuring large-scale annotations still remains expensive and time-intensive; with the problem even more acute for low-resource domains. This predicament necessitates focus on zero and few-shot methods for structured prediction.

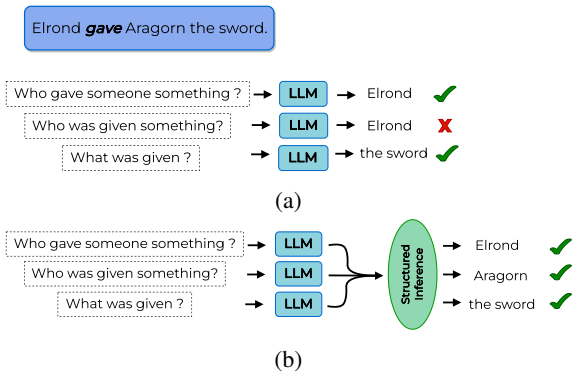

Figure 1: Example of Question Answer driven Semantic Role Labeling (QA-SRL) (a) without, and (b) with structured inference. Sans inference, prediction for each question may contain overlapping/repeated answers, which is prohibited per the task definition. Structured inference avoids such structurally invalid outputs.

Recently, prompt-based methods (e.g., Schick and Schütze, 2021a,b; Liu et al., 2023, inter alia) have shown immense promise, requiring few or even no labeled examples for competitive performance. Prompt-based models generate output text conditionally based on textual input called *prompts*. Prompts can be crafted using a pre-defined task-specific template (Raffel et al., 2020), or even just as natural language instructions (Chung et al., 2022) or questions (Gardner et al., 2019; Pyatkin et al., 2021).

Prompt-based methods have shown promise for

text classification and text generation tasks. However, their use for structured prediction is hitherto largely unexplored. Prior work (e.g., see Liu et al., 2023) uses prompts to predict components of structures independently, but their interactions are ignored. Figure 1a illustrates a failure of that agenda. Each question pertains to a semantic role for the highlighted predicate. Since the model independently encounters each question, it runs the risk of predicting structurally inconsistent answers.

In this paper, we propose to reintroduce inference into prompt-based methods for structured prediction problems (Figure 1b). We present a framework that allows using well-studied inference algorithms with prompt-based predictions. Inference algorithms—realized as beam search, integer linear programs (ILPs, e.g., Roth and Yih, 2004), weighted graph optimization (e.g., Täckström et al., 2015), etc.—take scored candidate label sub-structures as input and optimize a global score while satisfying structural constraints. Historically, trained models generated the label scores (e.g., Lafferty et al., 2001; Collins, 2002). We argue that large language models (LLMs), with appropriate prompts, can be used for scoring, thus eliminating the need for any training or fine-tuning.

We instantiate this idea on two English structured prediction tasks across five datasets. We explore various facets of prompt engineering (e.g., zero- vs few-shot, instruction tuning, etc.) and their impacts on performance. We show that current models can produce structurally inconsistent outputs, and inference helps improve not only consistency, but also overall task performance.[1]

2 Related Work

Prompts for Predictions. The "pretrain, prompt, and predict" paradigm (cf. Liu et al., 2023) exploits the observation that LLMs yield general-purpose tools applicable to several natural language tasks. These LLMs (e.g., T5 (Raffel et al., 2020), GPT-3 (Brown et al., 2020)) may be pretrained for various objectives, including the use of labeled data transformed into prompt-answer pairs. Wei et al. (2022) introduced *instruction tuning* that transforms natural language tasks into natural language instructions to generalize across tasks independent of their presence in training. With fine-tuning still possible, the prompting paradigm allows for generalization across tasks with some (few-shot) or even no (zero-shot) labeled samples. Several works have shown promise on zero- and few-shot text classification and generation tasks (e.g., Raffel et al., 2020; Brown et al., 2020; Schick and Schütze, 2021b; Lu et al., 2022; Touvron et al., 2023, inter alia).

Structures & Prompts. Many NLP tasks—e.g., parsing, coreference resolution, information extraction—call for predicting structured outputs such as sequences, trees or graphs. Structured prediction has a rich history in NLP (Smith, 2010).

Predicting structures with prompts is challenging; the output corresponds to multiple mutually-constraining decisions. Structures that can be flattened into label sequences can be cast as text generation tasks amenable to the LLM interface. Blevins et al. (2023) present a *structured prompting* paradigm to generate labels in an auto-regressive fashion for part-of-speech tagging, named entity recognition, and sentence chunking tasks. They show that this approach works well in a few-shot setting. Such tasks are also amenable for templated slot filling in a few shot setting (Cui et al., 2021). Liu et al. (2022) propose an autoregressive structured prediction framework that reformulates the problem into predicting a series of structure-building action steps.

Structured prediction tasks can be made prompt-friendly by identifying output components and framing each one into natural instructions (He et al., 2015; Du and Cardie, 2020; Pyatkin et al., 2020; Klein et al., 2022; Pyatkin et al., 2023) using the QA format. Mekala et al. (2023) decompose semantic parsing into a series of questions. However, they do not use inference and instead, aggregate outputs sequentially. Yang et al. (2022) use QA prompts to classify pairs of entity mentions as coreferent or not without aiming to form entity clusters. The near contemporary work of Lin et al. (2023) presents a prompt-driven heuristic for event type classification. Wang et al. (2023) use a programming language-based prompt and generate approach for zero and few-shot event argument extraction. Recently, Le and Ritter (2023) show two techniques based on QA and document infilling for zero-shot coreference resolution that rely on recent LLMs' ability to ingest entire documents as input. The concurrent work of Rajaby Faghihi and Kordjamshidi (2024) enforces structural consistency at inference-time, incorporating local decisions from different models using various normalizer functions.

[1]Code available at https://github.com/utahnlp/prompts-for-structures

3 "Promptly" Predicting Structures

Problem Statement and Notation Given an input X (e.g., phrase, sentence, etc.), the goal of a conditional model is to characterize the distribution $P(Y|X)$ over outputs Y. The outputs Y in structured prediction tasks consist of a set of decisions $y_1, y_2, \cdots$. For instance, consider the SRL task from §1. Given a predicate, identifying the token spans correponding to its arguments requires multiple decisions; i.e., each y_i corresponds to a semantic argument.

Since the size of the output Y is variable and depends on the input X, the standard approach for modeling such problems calls for factorizing the distribution $P(Y|X)$ over its components $y_1, y_2, \cdots$. The factorization is a design choice; the simplest one decomposes the probability as a product over individual local decisions: $P(Y|X) = \prod_i P(y_i|X)$. With such a factorization, the prediction Y^* for an input X is given by

$$Y^* = \max_Y \prod_{y_i \in Y} P(y_i \mid X) \tag{1}$$

These local probabilities relate to unary potentials in factor graph notation, and in our work, will correspond to logits from neural models. Importantly, the maximization problem above is trivial to solve. Each y_i can be independently assigned.[2]

Even the simple factorization above presents two drawbacks: i) It requires an explicit training step using expensive-to-acquire training data to obtain the unary potentials. ii) It does not guarantee structural consistency, leading to invalid structures. In this work, we ask: *Can we predict valid structured outputs in the zero- to few-shot regime?*

Now, let us address these drawbacks, leading to our proposed framework.

Unary Potentials from Prompts. Prompt-based methods can help bypass the need for supervised training. For every local decision y_i, we construct a query q_i to convert the decision into a prompt-friendly format. A pretrained model can use these queries in the standard prompt-and-predict fashion to get scored candidate answers. Formally, we write $P(Y|X,Q) = \prod_i P(y_i|X,q_i)$. Here, Q is a set of questions corresponding to the components of the structure. In the case of SRL, X would be the input sentence and the predicate, and Q would contain one question per semantic argument type for that predicate. Figure 1 gives an example.

The Return of Inference. Merely aggregating outputs generated with prompts does not address the second drawback, i.e., the need for structurally valid outputs. Structural validity is a constraint that, when applied to the prediction problem, ensures that only valid output collections are considered as candidate predictions. In other words, instead of the unconstrained prediction of Eq. 1, we have

$$Y^* = \max_Y \prod_{y_i \in Y} P(y_i|X,q_i) \quad \text{s.t. Y is structurally consistent} \tag{2}$$

The inference problem above requires combinatorial search over valid structures. Inference can be realized algorithmically in various forms like beam search, an ILP (Roth and Yih, 2004) or weighted graph search (e.g., Täckström et al., 2015; Zaratiana et al., 2022).

The Framework. We propose the following blueprint for predicting valid structures in a zero-shot manner: i) Break the prediction task into its constituent sub-tasks. ii) Convert each sub-task into a prompt-friendly format. iii) Implement an inference algorithm that takes scores from the LLM and produces a valid structured output. This recipe naturally extends to a few-shot setting by prefixing in-context examples with the original prompt.

Inference constructs outputs that satisfy constraints inherent in the definition of a task. Moreover, it can help correct errors of the zero- or few-shot models using other predictions of the same model. Without this mechanism (i.e., using the labels from the prompts directly as in Equation 1), not only will we have partially wrong outputs, they may be invalid from the perspective of the task. Finally, as we will see in the case of the coreference task, inference can help make long-range decisions about parts of an input even when the entire input is not presented to an LLM. In this sense, it can be seen as a mechanism to "bring back the context" without explicitly having to use a long-document model such as LongFormer (Beltagy et al., 2020).

[2] Of course, the structured prediction literature studies other task-dependent factorizations. With neural models, autoregressive factorizations are also common, where each label is conditioned on previously predicted ones. In this work, we study the simple factorization described here because it can be easily adapted to work with prompting.

4 Experiments

This section describes the tasks and datasets where we instantiate the framework from section §3.

4.1 Semantic Role Labeling

Semantic Role Labeling (SRL) is the task of constructing predicate-argument structures triggered by verbs in a sentence. This task has been studied in using the PropBank (Palmer et al., 2005), FrameNet (Baker et al., 1998), and more recently, the QA-SRL (He et al., 2015) datasets.

Datasets. The QA-SRL project (He et al., 2015) recasts the argument labeling task as question-answering to make it amenable for non-expert annotation. By breaking the predicate-argument structure into a collection of individual arguments, the project curated a collection of QA pairs that represent SRL frames. This SRL-to-QA transformation sets it up perfectly for prompting; the sentence along with each question can serve as a prompt.

We consider two QA-SRL datasets for our experiments. 1) **QA-SRL**$_{wiki}$ (He et al., 2015) contains ∼10.8k QA pairs for 4440 predicates in 1959 sentences across splits.[3] 2) FitzGerald et al. (2018) present a much larger crowd-sourced dataset—**QA-SRL 2.0**—containing 265k QA pairs from 64k sentences for 133k verbal predicates annotated using the QA-SRL annotation schema.

Constraints. The SRL definition dictates that different argument spans for a predicate do not have token overlap (Palmer et al., 2010), and the QA-SRL schema mandates an answer per question.

Prompts and Inference. For any predicate in a sentence, we have one question q_i per semantic role i. For every question, we prompt a generative language model to predict the best-n spans (i.e., sequences) with their respective scores. We denote the score from the language model for the r^{th} (where r $\leq$ n) ranked span (t_r) for role i as $f(t_r, i)$. The scored spans from all questions for a predicate are input to an inference algorithm.

We follow the inference algorithm of Täckström et al. (2015).[4] Given a sentence, we construct a directed graph with one more vertex than the number of tokens. (Figure 2 shows an example.) A vertex v_j is shared by two consecutive words w_{j-1} and w_j, and denotes the boundary between them. Edges can be of two kinds: i) Null edges are added between consecutive nodes with a zero edge score. ii) For every candidate span t_r ranked r for the role i, we add an edge between its representative vertices with an edge score $s_{r,i} = f(t_1, i) - f(t_r, i)$. The highest ranked span gets an edge score of zero.[5]

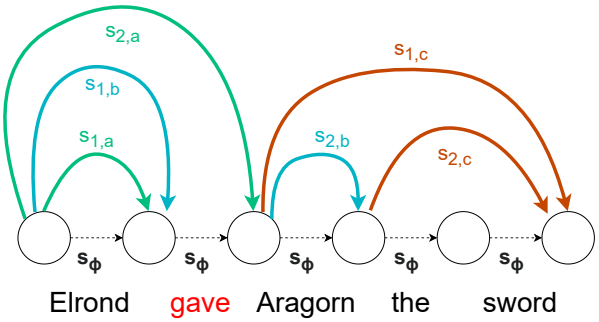

Figure 2: An example graph for the statement "*Elrond gave Aragorn the sword*". This is a toy example with the top-2 candidate spans ($n = 2$) for semantic roles=$\{a, b, c\}$. Each edge represents a candidate span. For instance, the span ranked second for scene role 'a' is the phrase "*Elrond gave*" with an edge score of $s_{2,a}$.

Given the graph, we can perform a shortest-path search from the leftmost to the rightmost nodes; the edges in the shortest path give us non-overlapping labels. However, doing so will generate null sequences that have a zero score. We avoid this issue by extending the algorithm to return the top K shortest paths using Yen's K-shortest path algorithm (Yen, 1971). Subsequently, the highest scoring path among the K paths which assigns exactly one span per semantic role is the optimal structure that satisfies all task constraints. For the example in Figure 2, our inference algorithm would return '*Elrond*', '*Aragorn*', and '*the sword*' as the arguments for roles a, b and c respectively.[6]

Evaluation. We evaluate at both the question-level and the structure-level. At each level, we use exact and head match metrics (Palmer et al., 2010, p. 49). At the question-level, predicted spans are compared with gold spans. For the **Exact**$_q$ accuracy metric, answers are considered correct if the spans exactly match. Similarly, answers are considered correct if the phrasal heads of the spans match to compute the **Head**$_q$ accuracy metric. A structure

[3] He et al. (2015) provide datasets in newswire and Wikipedia domains. We use the latter as it is larger.

[4] For brevity, we outline the inference setup here, and refer the reader to the original work for a detailed description.

[5] A generative model tasked for extraction, like ours, cannot get indices to mark the start and end. Yet, a generated span may occur multiple times in a sentence. In such cases, edges representing all such spans are added with an identical score.

[6] See Appendix F for a worked-out example.

is considered accurate if predicted spans for all constituent roles match. As at the question-level, a match can be determined in an exact (**Exact**$_s$), or head (**Head**$_s$) fashion. Additionally, we also evaluate inconsistency percent (ρ) by measuring how often constraints are violated. We compare every pair of outputs(i.e., argument spans) in a predicted structure. A violation occurs when two output spans overlap.

4.2 Coreference Resolution

Coreference resolution is the task of grouping mention expressions (or simply, mentions) in a document into clusters that refer to the same entity. For example, consider the sentence "*Al requested Bob to give him the pen.*", with given mentions: {"Al", "Bob", "him", "the pen"}. Here, "Al" and "him" refer to the same entity, i.e., Al, while the other mentions are singletons.

Datasets. We use three datasets for this task: 1) The entity coreference annotations within the EventCorefBank+ dataset (ECB+, Cybulska and Vossen, 2014).[7] 2) The CoNLL 2012 OntoNotes 5.0 dataset with the v12 English data splits (Pradhan et al., 2012), which contains entity and event coreference annotation across several domains like magazine articles, broadcast conversations, among others. 3) The GENIA Coreference Corpus (Su et al., 2008), which helps showcase the applicability of this framework in a low-resource domain like biomedicine where annotation is especially difficult. The GENIA corpus is a collection of coreference annotations for 2000 paper abstracts published in the biomedical domain. We define our own train/dev/test splits for the corpus since pre-defined splits were unavailable. We will release these splits for future use. For OntoNotes and GENIA, we do not consider nested entities.

Constraints. The coreference resolution task can be broken into a series of binary decisions: for every mention pair in a document we can ask whether the mentions are co-referrant or not. Entity clusters require transitive closure. That is, given three mentions m_i, m_j and m_k, if a predicate $C(x, y)$ denotes x and y refer to the same entity, then:

$$\forall i, j, k, C(m_i, m_j) \wedge C(m_j, m_k) \implies C(m_i, m_k) \quad (3)$$

[7]We use the processed files provided by Yang et al. (2022)

Prompt and Inference. For every document, we collect mention pairs in it.[8] We transform the coreference decision about mentions m_i and m_j into a Yes/No question: "*Does m_i refer to m_j?*". A 'Yes' establishes a coreference link between the mentions. Questions are appended with the relevant context and given to a language model. Link scores ($s_{i,j}$) for a mention-pair i and j are given by $s_{i,j} = f_{yes}(i, j) - f_{no}(i, j)$ where f_{yes} and f_{no} are the scores for generating the sequences "yes" and "no" given a question and context. Using the link scores, we employ *All-link* inference from Chang et al. (2011) to solve the following integer program over binary decisions $y_{i,j}$:

$$\max_y \sum_{i,j} y_{i,j} s_{i,j}$$
$$\text{such that} \quad y_{i,k} \geq y_{i,j} + y_{j,k} - 1, \quad \forall i, j, k$$
$$y_{i,j} \in \{0, 1\}, \quad \forall i, j.$$

The constraint ensures that the transitivity condition from (3) is satisfied.

Evaluation. We use two metrics: i) **F1** that measures performance of the binary decisions at the question level ii) **CoNLL score** (Pradhan et al., 2014) that averages three cluster evaluation metrics: MUC , B^3, and $CEAF_e$. Additionally, we measure the inconsistency percent (ρ): the number of times constraint (3) is violated divided by the number of times the antecedent is true. This is the same as the 'conditional violation' defined by Li et al. (2019), and measures the fraction of times a third link between three mentions does not exist, when the other two links between them exist.

5 Results

This section presents the impact of inference on zero-shot performance and consistency. We use the 3 billion variants from the T5 (Raffel et al., 2020) family of encoder-decoder models for our experiments. We discuss our implementation briefly here; Appendix D has details like prompt templates.

5.1 Semantic Role Labeling

We consider the T5-3B model[9] for our experiments. We also present results on Flan T5-XL (Wei et al., 2022), also a 3B parameter model, to check if instruction tuning helps with zero-shot performance.

[8]For Ontonotes, we considered all mention pairs in a window size of three sentences due to a large number of mentions.

[9]We also considered UnifiedQAv2 (Khashabi et al., 2022) and Macaw (Tafjord and Clark, 2021). T5 performed the best.

Instruction-tuned models like Flan T5 allow verbalizing constraints within the prompt structure. We also test out an iterative prompting method (Wang et al., 2022), where questions pertaining to a predicate in a sentence are presented in sequential order to the model with verbalized constraints. In every prompt, we include the QA pair from every previous question for the predicate already prompted. This prompting strategy is denoted by the superscript `itr`. See Appendix D.1.1 for an example.[10] Further, we can use inference in conjunction with this iterative prompting strategy.

Tables 1 and 2 show the results for QA-SRL$_{wiki}$ and QA-SRL 2.0 respectively. We see that the constrained models produce near-consistent gains over their unconstrained counterparts. Flan-T5 does not outperform the T5 model, but iterative prompting with inference significantly improves its performance. More importantly, without inference, the models produce highly inconsistent predictions with a lowest inconsistency percent of 34%! In addition, we also provide results on GPT-4 with the constraints verbalized as system instruction.[11] For GPT-4, all the questions are provided at once. Refer Appendix D.1.2 for details.

As an illustrative example, we present a case where constrained inference corrects overlapping predicted arguments. For the sentence, *"Keats' long and expensive medical training with Hammond and at Guy's Hospital led his family to assume he would pursue a lifelong career in medicine, assuring financial security ..."*, the model was asked: i) What would assure something? ii) What would something assure? The unconstrained model generated "financial security" on both counts. Post-inference, the answer to the first question is changed to "lifelong career in medicine", resulting in the correct structure.

5.2 Coreference Resolution

The ability to specify multiple choices makes the Macaw-3B (Tafjord and Clark, 2021) model an ideal choice for the Yes/No choice QA task.[12] As before, we show results using the instruction-tuned Flan T5-XL model as well. For this set of experiments, we only prompt the model with the sentence(s) containing the mention pair as context, and not the whole document.[13] We show more analysis with varying context styles in Appendix B.2.

In addition to the constrained inference proposed in §4.2, we consider a simple right-to-left cluster assignment heuristic (**R2L**) proposed by Soon et al. (2001). This heuristic assigns a candidate mention to the cluster of the closest mention where a link was predicted by the unconstrained model. We also present two rudimentary baselines: i) **All-Yes** baseline where all questions are answered "Yes" and all mentions belong to a single cluster, and ii) **All-No** baseline where all questions are answered "No" and every mention is its own cluster. These baselines are relevant since both generate valid structures.

The unconstrained and the constrained models can only be compared using their F1 scores and the inconsistencies.[14] Tables 3, 4 and 5 show results

System	Head$_q$(↑)	Head$_s$(↑)	ρ (↓)
T5-3B	**63.84**	36.17	34.30
+ constraints	62.11	**38.75**	0
Flan-T5-XL	36.76	11.20	87.16
+ constraints	43.34	18.48	0
Flan-T5-XL^{itr}	59.47	33.60	49.79
+ constraints	61.11	36.17	0
GPT-4	80.46	62.93	5.61

Table 1: Semantic Role Labeling performance and consistency metrics on the QA-SRL$_{wiki}$ dataset for unconstrained vs constrained systems. A fine-tuned constrained T5 model gets a Head$_s$ of 58.12. All values in %. Detailed metrics in Table 6 in Appendix A.

System	Head$_q$(↑)	Head$_s$(↑)	ρ (↓)
T5-3B	68.57	46.35	39.06
+ constraints	68.87	52.31	0
Flan-T5-XL	61.92	40.37	58.19
+ constraints	66.02	48.68	0
Flan-T5-XL^{itr}	68.84	49.29	42.11
+ constraints	**71.97**	**55.41**	0
GPT-4	86.78	76.89	7.18

Table 2: Semantic Role Labeling performance and consistency metrics on the QA-SRL 2.0 dataset for unconstrained vs constrained systems. A fine-tuned constrained T5 model gets a Head$_s$ of 69.00. All values in %. Detailed metrics in Table 7 in Appendix A.

[10] Note that verbalized constraints do not guarantee a valid output structure. However, exposure to preceding predictions may help reduce inconsistencies nonetheless.

[11] GPT-4 results are meant to contextualize our results with a current SOTA system. The datasets we considered may have been utilized to train GPT-4. GPT-4 also has multiple orders of magnitude more parameters than the models we consider.

[12] We also considered this task in an NLI format with the T5-3B model. However, the QA format was a clear winner.

[13] Note: We do not need a large context window. Inference can create consistent clusters by looking at smaller contexts.

[14] The official CoNLL scorer requires valid structures ren-

System	F1(↑)	CoNLL(↑)	ρ(↓)
All-Yes	13.32	41.51	0
All-No	45.84	45.06	0
Macaw-3B	46.26	N/A	48.56
+ R2L	42.20	53.55	0
+ All-Link	48.57	57.76	0
Flan-T5-XL	61.58	N/A	80.78
+ R2L	58.38	62.79	0
+ All-Link	**66.06**	**65.09**	0

Table 3: Coreference resolution performance and consistency metrics for unconstrained vs constrained systems for the ECB+ dataset. A fine-tuned constrained Macaw model gets an F1 and CoNLL score of 85.69 and 85.14 respectively. All values in %.

System	F1(↑)	CoNLL(↑)	ρ(↓)
All-Yes	18.96	38.61	0
All-No	43.38	39.42	0
Macaw-3B	48.67	N/A	63.88
+ R2L	49.48	**55.30**	0
+ All-Link	**52.15**	55.14	0
Flan-T5-XL	48.93	N/A	86.75
+ R2L	51.88	50.37	0
+ All-Link	51.33	48.52	0

Table 4: Coreference resolution performance and consistency metrics for unconstrained vs constrained systems for the OntoNotes dataset. A fine-tuned constrained Macaw model gets an F1 and CoNLL score of 96.17 and 95.63 respectively. All values in %.

for the ECB+, OntoNotes and GENIA datasets respectively. We refer the reader to Table 8 in Appendix B.1 for a detailed breakdown of the CoNLL score. We observe gains in F1 scores for constrained systems over their unconstrained counterparts. In all cases, our models outperform the All-Yes and All-No baselines. We also see that the inconsistency is substantially higher for the unconstrained models, especially with Flan-T5-XL.

Figure 3 presents an instance where inference corrects flawed links. Here, '*T-mobile*' and '*carrier*' belong to one entity cluster, while, '*Blackberry Curve 8900*' and '*product*' belong to another. Our unconstrained model incorrectly predicts a link between '*T-mobile*' and '*Blackberry Curve 8900*'. But transitivity requires all other edges to be present, which the model does not predict. Our inference algorithm removes the erroneous edge and predicts consistent clusters.

dering base system evaluation impossible w/o inference.

System	F1(↑)	CoNLL(↑)	ρ(↓)
All-Yes	9.00	37.70	0
All-No	47.40	31.19	0
Macaw-3B	46.63	N/A	51.18
+ R2L	35.62	47.25	0
+ All-Link	46.75	52.56	0
Flan-T5-XL	56.53	N/A	82.64
+ R2L	53.35	54.11	0
+ All-Link	**65.36**	**57.47**	0

Table 5: Coreference resolution performance and consistency metrics for unconstrained vs constrained systems for the GENIA dataset. A fine-tuned constrained Macaw model gets an F1 and CoNLL score of 91.58 and 90.01 respectively. All values in %.

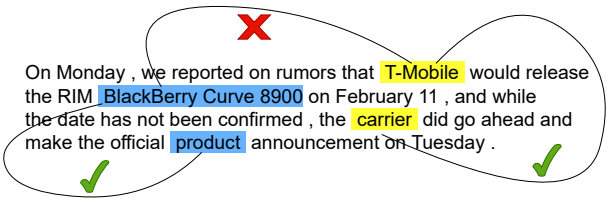

Figure 3: An example from ECB+ dataset where inference helps correct inconsistency of predictions. An incorrect link is predicted between '*Blackberry Curve 8900*' and '*T-Mobile*' by the unconstrained model, which is removed post-inference. Irrelevant mentions are hidden for clarity.

Iterative prompting for coreference resolution is challenging for two reasons: 1) As the component questions establish links between two mentions and the transitivity closure applies to three mentions, verbalizing the transitivity constraint within the prompt can be challenging. 2) Coreference structures are much larger than the SRL structures, hence, demanding a large memory requirement.

6 Analysis Experiments

Next, we show how performance and inconsistency vary with model size and in-context examples.

6.1 Model Size

We study the performance and consistency of models over varying sizes. For SRL, we compare the small, base, large, 3 billion, and 11 billion T5 models. Figure 4 shows the Head_s metric over T5 model sizes for QA-SRL 2.0, along with the inconsistency percent. For coreference, we compare the large, 3 billion and 11 billion variants of the Macaw model. We consider the All-Link inference for the constrained models. We report the F1 score

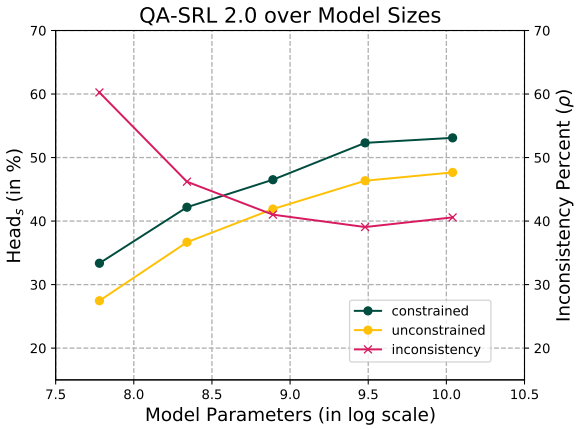

Figure 4: Head$_s$ performance and inconsistency percent over model sizes for the QA-SRL 2.0 dataset. The circle-marked (-•-) plots map to the left axis, and the cross-marked (-x-) plot to the right axis. Inconsistency is shown for unconstrained models since constrained models are always consistent (i.e., ρ=0).

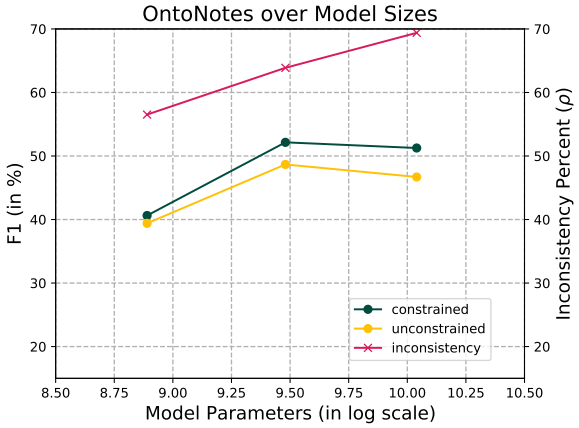

Figure 5: F1 performance and inconsistency percent over model sizes for the OntoNotes dataset. The visual encodings follow the same convention as in Figure 4.

and the inconsistency percent on the OntoNotes dataset in Figure 5. The inconsistency is shown only for the unconstrained models; the constrained ones are guaranteed to be consistent.

Across the datasets, we see that performance almost always increases and constraints provide steady gains. In QA-SRL 2.0, inconsistency reduces with increasing model size. The trend reverses for OntoNotes. Interestingly, with QA-SRL 2.0, a constrained model of a certain size often yields comparable, if not better, results as an unconstrained model of the immediately bigger model, showing that constraints might give performance gains equivalent to using a bigger model. Appendix C shows results for the other datasets.

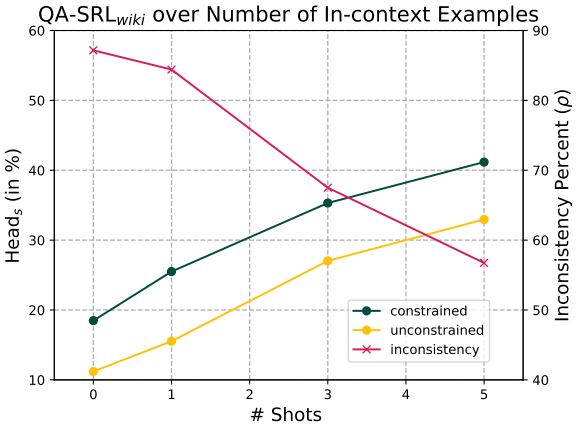

Figure 6: Head$_s$ performance and inconsistency percent over in-context examples (shots) for the QA-SRL$_{wiki}$ dataset. The visual encodings follow the same convention as in Figure 4.

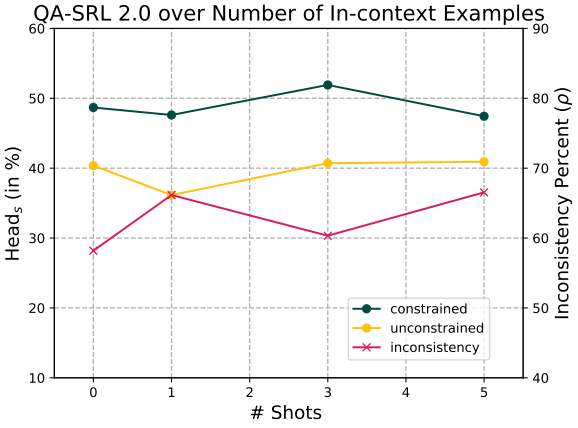

Figure 7: Head$_s$ performance and inconsistency percent over in-context examples (shots) for the QA-SRL 2.0 dataset. The visual encodings follow the same convention as in Figure 4.

6.2 Few-shot Prompting

So far, we have examined our framework in the zero-shot setting. *Do the conclusions hold for few-shot settings as well?* To answer this question, we use the Flan-T5-XL model which is known to yield promising results with even a few in-context examples.[15] We conduct few-shot experiments for the SRL task on the QA-SRL$_{wiki}$ (Figure 6) and QA-SRL 2.0 (Figure 7). We observe a clear increase in performance with increasing shots for QA-SRL$_{wiki}$. Recall that Flan T5, in a zero-shot setting for this dataset, underperforms compared to T5. Hence, the gains are not surprising: only the

[15]An interesting question: What counts as a shot? All questions in a structure or just one? We consider the latter.

5-shot constrained system beats the zero-shot constrained T5 model (38.75 Head_s). The model also becomes more consistent with more examples. For QA-SRL 2.0, the performance and the consistency largely remain invariant with number of shots. Constrained systems produce more accurate structures than their unconstrained counterparts.

7 Conclusions

Prompt-based methods have impressed when deployed in zero- and few-shot manner. Structured prediction problems can use them by breaking the structure down into smaller components and querying for local decisions. This aspect has been explored by some in the literature. However, these works leave out a pivotal part of such structured prediction tasks—the structure. In this work, we proposed a new framework to predict structurally consistent outputs by following local predictions with an inference step. In our experiments, our framework not only guarantees consistency but also provides consistent performance gains.

Limitations

Our focus of this work is to predict structures given that a structured prediciton task can be broken down into components and questions can be generated for those components. While this relies heavily on advances in question generation, we do not focus on this line of research and consider that questions are given to us. A separate line of work exists precisely in this domain of question generation for structured prediction tasks (He et al., 2015; Levy et al., 2017; Du and Cardie, 2020; Wu et al., 2020; Pyatkin et al., 2020; Klein et al., 2020; Pyatkin et al., 2021). We emphasize that our contribution is more general as to how we can marry the idea of inference with prompting.

A problem that can affect any generative model is the label bias problem where the model prefers a certain label over certain other label irrespective of the inputs. This problem can cause performance imbalances and models need to be calibrated to alleviate this phenomenon. Recently, a few works (Zhao et al., 2021; Holtzman et al., 2021; Chen et al., 2023; Han et al., 2023; Fei et al., 2023) study this phenomenon for zero and few shot prompting models for an array of text classification tasks. We briefly studied this for our coreference experiments and found that Macaw was more well-calibrated for the binary task as compared to T5. We leave a detailed calibration study for future work.

Acknowledgements

We thank the members at the Utah NLP lab and AI2 for their insights, especially, Ashim Gupta for his feedback on the iterative prompting experiments. We thank the anonymous reviewers for their feedback. This material is based upon work supported in part by NSF under grants #2007398, #2217154, #1822877 and #1801446. The support and resources from the Center for High Performance Computing at the University of Utah are gratefully acknowledged. Valentina is supported by an Eric and Wendy Schmidt Postdoctoral Award.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv preprint arXiv:2004.05150*.

Terra Blevins, Hila Gonen, and Luke Zettlemoyer. 2023. Prompting Language Models for Linguistic Structure. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6649–6663, Toronto, Canada. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models Are Few-Shot Learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Kai-Wei Chang, Rajhans Samdani, Alla Rozovskaya, Nick Rizzolo, Mark Sammons, and Dan Roth. 2011. Inference Protocols for Coreference Resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 40–44, Portland, Oregon, USA. Association for Computational Linguistics.

Yangyi Chen, Lifan Yuan, Ganqu Cui, Zhiyuan Liu, and Heng Ji. 2023. A Close Look into the Calibration

of Pre-trained Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1343–1367, Toronto, Canada. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. *arXiv preprint arXiv:2210.11416*.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-Based Named Entity Recognition Using BART. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Using a Sledgehammer to Crack a Nut? Lexical Diversity and Event Coreference Resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland. European Language Resources Association (ELRA).

Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Yu Fei, Yifan Hou, Zeming Chen, and Antoine Bosselut. 2023. Mitigating Label Biases for In-context Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14014–14031, Toronto, Canada. Association for Computational Linguistics.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-Scale QA-SRL Parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2051–2060, Melbourne, Australia. Association for Computational Linguistics.

Matt Gardner, Jonathan Berant, Hannaneh Hajishirzi, Alon Talmor, and Sewon Min. 2019. Question Answering is a Format; When is it Useful? *arXiv preprint arXiv:1909.11291*.

Gurobi Optimization, LLC. 2023. Gurobi Optimizer Reference Manual.

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. 2023. Prototypical Calibration for Few-shot Learning of Language Models. In *Proceedings of the Eleventh International Conference on Learning Representations*.

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, Lisbon, Portugal. Association for Computational Linguistics.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface Form Competition: Why the Highest Probability Answer Isn't Always Right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daniel Khashabi, Yeganeh Kordi, and Hannaneh Hajishirzi. 2022. UnifiedQA-v2: Stronger Generalization via Broader Cross-Format Training. *arXiv preprint arXiv:2202.12359*.

Ayal Klein, Eran Hirsch, Ron Eliav, Valentina Pyatkin, Avi Caciularu, and Ido Dagan. 2022. QASem Parsing: Text-to-text Modeling of QA-based Semantics. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7742–7756, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ayal Klein, Jonathan Mamou, Valentina Pyatkin, Daniela Stepanov, Hangfeng He, Dan Roth, Luke Zettlemoyer, and Ido Dagan. 2020. QANom: Question-Answer driven SRL for Nominalizations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3069–3083, Barcelona, Spain (Online). International Committee on Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Nghia T Le and Alan Ritter. 2023. Are Large Language Models Robust Zero-shot Coreference Resolvers? *arXiv preprint arXiv:2305.14489*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A Logic-Driven Framework for Consistency of Neural Models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3924–3935, Hong Kong, China. Association for Computational Linguistics.

Zizheng Lin, Hongming Zhang, and Yangqiu Song. 2023. Global Constraints with Prompting for Zero-Shot Event Argument Classification. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 2527–2538, Dubrovnik, Croatia. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Computing Surveys*, 55(9):1–35.

Tianyu Liu, Yuchen Eleanor Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. 2022. Autoregressive Structured Prediction with Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Dheeraj Mekala, Jason Wolfe, and Subhro Roy. 2023. ZEROTOP: Zero-Shot Task-Oriented Semantic Parsing using Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5792–5799, Singapore. Association for Computational Linguistics.

S. Nowozin, P.V. Gehler, J. Jancsary, and C.H. Lampert. 2014. *Advanced Structured Prediction*. Neural Information Processing series. MIT Press.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. Semantic Role Labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring Coreference Partitions of Predicted Mentions: A Reference Implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–35, Baltimore, Maryland. Association for Computational Linguistics.

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, Jeju Island, Korea. Association for Computational Linguistics.

Valentina Pyatkin, Ayal Klein, Reut Tsarfaty, and Ido Dagan. 2020. QADiscourse - Discourse Relations as QA Pairs: Representation, Crowdsourcing and Baselines. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2804–2819, Online. Association for Computational Linguistics.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking It All: Generating Contextualized Questions for any Semantic Role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Valentina Pyatkin, Frances Yung, Merel C. J. Scholman, Reut Tsarfaty, Ido Dagan, and Vera Demberg. 2023. Design Choices for Crowdsourcing Implicit Discourse Relations: Revealing the Biases Introduced by Task Design. *Transactions of the Association for Computational Linguistics*, 11:1014–1032.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Hossein Rajaby Faghihi and Parisa Kordjamshidi. 2024. Consistent Joint Decision-Making with Heterogeneous Learning Models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 803–813, St. Julian's, Malta. Association for Computational Linguistics.

Dan Roth and Wen-tau Yih. 2004. A Linear Programming Formulation for Global Inference in Natural Language Tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021a. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. In *Proceedings of the 2021*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2339–2352, Online. Association for Computational Linguistics.

Noah A. Smith. 2010. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4):521–544.

Jian Su, Xiaofeng Yang, Huaqing Hong, Yuka Tateisi, and Jun'ichi Tsujii. 2008. Coreference Resolution in Biomedical Texts: a Machine Learning Approach. In *Ontologies and Text Mining for Life Sciences : Current Status and Future Perspectives*, volume 8131 of *Dagstuhl Seminar Proceedings (DagSemProc)*, pages 1–1, Dagstuhl, Germany. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

Oyvind Tafjord and Peter Clark. 2021. General-Purpose Question-Answering with Macaw. *ArXiv*, abs/2109.02593.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient Inference and Structured Learning for Semantic Role Labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41.

Boshi Wang, Xiang Deng, and Huan Sun. 2022. Iteratively Prompt Pre-trained Language Models for Chain of Thought. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2714–2730, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xingyao Wang, Sha Li, and Heng Ji. 2023. Code4Struct: Code Generation for Few-Shot Event Structure Prediction. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3640–3663, Toronto, Canada. Association for Computational Linguistics.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned Language Models are Zero-Shot Learners. In *Proceedings of the Tenth International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference Resolution as Query-based Span Prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6953–6963, Online. Association for Computational Linguistics.

Xiaohan Yang, Eduardo Peynetti, Vasco Meerman, and Chris Tanner. 2022. What GPT Knows About Who is Who. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 75–81, Dublin, Ireland. Association for Computational Linguistics.

Jin Y Yen. 1971. Finding the K Shortest Loopless Paths in a Network. *Management Science*, 17(11):712–716.

Urchade Zaratiana, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2022. Named Entity Recognition as Structured Span Prediction. In *Proceedings of the Workshop on Unimodal and Multimodal Induction of Linguistic Structures (UM-IoS)*, pages 1–10, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate Before Use: Improving Few-shot Performance of Language Models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

A Detailed SRL Results

Detailed results for the QA-SRL$_{wiki}$ and QA-SRL 2.0 are given in Tables 6 and 7 respectively.

B Additional Coreference Results

B.1 Zero Shot Coreference Results

The complete zero shot results for Macaw-3B and Flan T5-XL models are given in Table 8. One might notice that the Flan model is highly inconsistent as compared to the Macaw model. Note that in each case, the divisor is different since it is the number of times the antecedent is activated in equation 3. Examining the predictions, we saw that the value of the divisor for the Macaw models is roughly four times that of the Flan models. This implies that Flan-T5 tends to predict 'No' more often.

System	$\text{Exact}_q(\uparrow)$	$\text{Exact}_s(\uparrow)$	$\text{Head}_q(\uparrow)$	$\text{Head}_s(\uparrow)$	$\rho\ (\downarrow)$
T5-3B	38.80	12.65	**63.84**	36.17	34.30
+ constraints	40.30	14.78	62.11	**38.75**	0
Flan-T5-XL	22.72	3.47	36.76	11.20	87.16
+ constraints	33.12	9.52	43.34	18.48	0
Flan-T5-XL^{itr}	37.57	12.32	59.47	33.60	49.79
+ constraints	**43.80**	**17.25**	61.11	36.17	0
GPT-4	62.84	37.07	80.46	62.93	5.61

Table 6: Semantic Role Labeling performance and consistency metrics on the QA-SRL$_{wiki}$ dataset for unconstrained vs constrained systems. A fine-tuned constrained T5 model gets a Head_s of 58.12. All values in %.

System	$\text{Exact}_q(\uparrow)$	$\text{Exact}_s(\uparrow)$	$\text{Head}_q(\uparrow)$	$\text{Head}_s(\uparrow)$	$\rho\ (\downarrow)$
T5-3B	54.12	30.49	68.57	46.35	39.06
+ constraints	56.78	36.96	68.87	52.31	0
Flan-T5-XL	48.92	26.53	61.92	40.37	58.19
+ constraints	57.06	37.23	66.02	48.68	0
Flan-T5-XL^{itr}	55.11	32.86	68.84	49.29	42.11
+ constraints	**61.34**	**41.17**	**71.97**	**55.41**	0
GPT-4	77.98	63.00	86.78	76.89	7.18

Table 7: Semantic Role Labeling performance and consistency metrics on the QA-SRL 2.0 dataset for unconstrained vs constrained systems. A fine-tuned constrained T5 model gets a Head_s of 69.00. All values in %.

Dataset	System	F1(↑)	MUC(↑)	$\text{B}^3(\uparrow)$	$\text{CEAF}_e(\uparrow)$	CoNLL(↑)	$\rho(\downarrow)$
ECB+	Single	13.32	**63.93**	41.67	18.94	41.51	0
	Singleton	45.84	0.00	74.13	61.05	45.06	0
	Macaw-3B	46.26	N/A	N/A	N/A	N/A	48.56
	+ R2L	42.20	52.71	61.00	46.93	53.55	0
	+ All-Link	48.57	56.72	65.59	50.98	57.76	0
	Flan-T5-XL	61.58	N/A	N/A	N/A	N/A	80.78
	+ R2L	58.38	53.03	73.27	62.08	62.79	0
	+ All-Link	**66.06**	51.59	**77.07**	**66.60**	**65.09**	0
Ontonotes	Single	18.96	**70.23**	30.89	14.71	38.61	0
	Singleton	43.38	0.00	66.43	51.84	39.42	0
	Macaw-3B	48.67	N/A	N/A	N/A	N/A	63.88
	+ R2L	49.48	51.11	66.04	48.76	**55.30**	0
	+ All-Link	**52.15**	47.67	67.02	50.72	55.14	0
	Flan-T5-XL	48.93	N/A	N/A	N/A	N/A	86.75
	+ R2L	51.88	31.12	**67.51**	52.47	50.37	0
	+ All-Link	51.33	24.93	67.08	**53.56**	48.52	0
GENIA	Single	9.00	**77.41**	26.81	8.89	37.70	0
	Singleton	47.40	0.00	57.10	36.47	31.19	0
	Macaw-3B	46.63	N/A	N/A	N/A	N/A	51.18
	+ R2L	35.62	66.00	45.11	30.65	47.25	0
	+ All-Link	46.75	62.40	56.19	39.08	52.56	0
	Flan-T5-XL	56.53	N/A	N/A	N/A	N/A	82.64
	+ R2L	53.35	59.62	58.30	44.41	54.11	0
	+ All-Link	**65.36**	53.48	**66.90**	**52.04**	**57.47**	0

Table 8: Coreference resolution performance and consistency metrics for unconstrained vs constrained systems. All values in %.

B.2 Context Styles for Coreference Resolution

As mentioned in §5.2, for any given question pertaining to a mention-pair, we consider only the sentences where the constituent mentions occur. One can argue that surrounding context can play an important role in deciding whether or not two mentions are co-referent. Furthermore, intermediate context between the sentences can be pivotal in establishing otherwise loosely connected sentences appearing a few sentences apart. This begs the question about how much, if any, does providing the entire document help.[16] On a different note, mentions can be highlighted with asterisks(as used for WSC in Raffel et al. (2020)) to explicitly state where in the context they occur. This approach can help deal with pronominal mentions which are bound to repeat. We present results when the entire document (**Full**) acts as context, when mentions are highlighted (**Hlght**), and a combination of both (**Full + Hlght**) in comparison to the context style where only the relevant sentences are considered (**Rel**). We provide detailed results for performance and consistency over differing context styles in Table 9. We see that results are comparable across all context styles.

C Additional Model Size Experiments

The analysis over model sizes for QA-SRL$_{wiki}$ is given in Figure 8. We see similar analysis as observed with QA-SRL 2.0 dataset. Performance improves with increasing model size while inconsistency reduces. Furthermore, constraints provide steady gains over their unconstrained counterparts.

Figure 9 shows performance and inconsistency of the Macaw model over sizes for the ECB+ coreference dataset. Figure 10 shows the same for the GENIA dataset. In both the cases, we see that performance drastically improves from the large to the 3 billion parameter variant. Performance gains from 3 billion to 11 billion are minimal. However, we see that the inconsistency increases with increasing size. We also see that for these datasets, for the large variant, constraints do not help the model. However, note that F1 is a question-level metric and a drop might be a necessary to guarantee a consistent structure.

[16]For Ontonotes and GENIA, the document size went beyond memory which is why we do not test along this dimension.

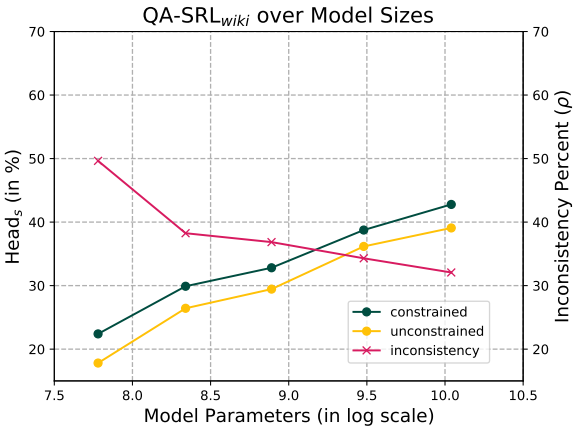

Figure 8: Head$_s$ performance and inconsistency percent over model sizes for the QA-SRL$_{wiki}$ dataset. Refer circle-marked (-•-) plots to the left axis, and the cross-marked (-x-) plot to the right axis. Inconsistency is computed for unconstrained models since constrained models are trivially consistent (i.e., ρ=0).

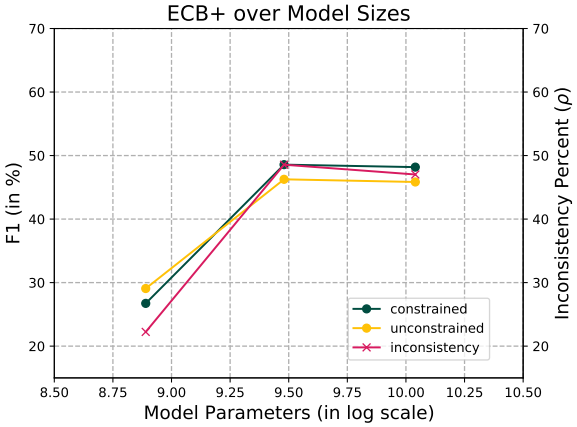

Figure 9: F1 performance and inconsistency percent over model sizes for the ECB+ dataset. Refer circle-marked (-•-) plots to the left axis, and the cross-marked (-x-) plot to the right axis. Inconsistency is computed for unconstrained models since constrained models are trivially consistent (i.e., ρ=0).

Dataset	System	F1(↑)	MUC(↑)	B^3(↑)	$CEAF_e$(↑)	CoNLL(↑)	ρ(↓)
ECB+	Rel	46.26	N/A	N/A	N/A	N/A	48.56
	+ R2L	42.20	52.71	61.00	46.93	53.55	0
	+ All-Link	**48.57**	56.72	**65.59**	**50.98**	**57.76**	0
	Hlght	46.03	N/A	N/A	N/A	N/A	46.68
	+ R2L	42.49	51.19	60.76	47.17	53.04	0
	+ All-Link	47.84	57.92	64.80	49.59	57.44	0
	Full	44.47	N/A	N/A	N/A	N/A	31.15
	+ R2L	38.46	57.51	60.25	43.97	53.91	0
	+ All-Link	45.59	59.28	64.69	48.93	57.63	0
	Full+Hlght	43.54	N/A	N/A	N/A	N/A	31.02
	+ R2L	37.38	59.17	59.44	43.15	53.92	0
	+ All-Link	44.66	**60.34**	63.80	48.56	57.57	0
Ontonotes	Rel	48.67	N/A	N/A	N/A	N/A	63.88
	+ R2L	49.48	51.11	66.04	48.76	55.30	0
	+ All-Link	52.15	47.67	67.02	50.72	55.14	0
	Hlght	49.22	N/A	N/A	N/A	N/A	64.93
	+ R2L	49.40	**51.16**	65.88	48.75	55.26	0
	+ All-Link	**52.43**	48.20	**67.13**	**50.90**	**55.41**	0
GENIA	Rel	46.63	N/A	N/A	N/A	N/A	51.18
	+ R2L	35.62	66.00	45.11	30.65	47.25	0
	+ All-Link	**46.75**	62.40	**56.19**	39.08	52.56	0
	Hlght	46.55	N/A	N/A	N/A	N/A	50.62
	+ R2L	34.94	**66.29**	45.30	30.81	47.47	0
	+ All-Link	46.66	62.67	55.92	**39.22**	**52.60**	0

Table 9: Coreference resolution performance and consistency metrics over different context styles for unconstrained vs constrained systems. The first row of every triplet denotes the context style considered. All values in %. All results are on the Macaw-3B model.

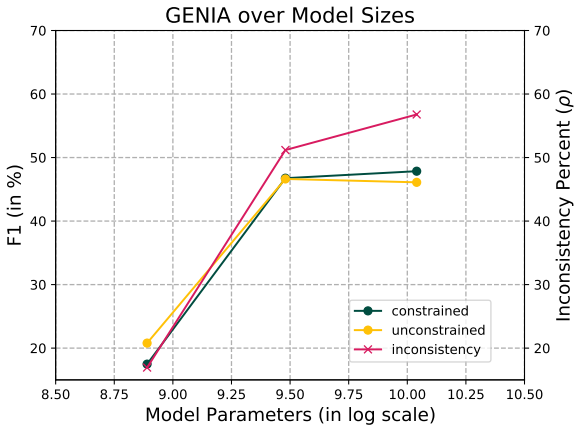

Figure 10: F1 performance and inconsistency percent over model sizes for the GENIA dataset. Refer circle-marked (-•-) plots to the left axis, and the cross-marked (-x-) plot to the right axis. Inconsistency is computed for unconstrained models since constrained models are trivially consistent (i.e., ρ=0).

D Experimental and Prompt Details

We use HuggingFace's `transformers` library (Wolf et al., 2020) for all our models. All model generations were done with a random seed of 2121. We use the 'generate' method in the HuggingFace (Wolf et al., 2020) library to return the score of a generated sequence. These scores can be accessed by the 'sequences_scores' output attribute. We consider the top 20 spans for each question. For all few shot settings, we average results across shots considered from three seeds: 42, 20 and 1984.

We explain the prompt templates and the parameters/assumptions we chose in the subsequent sections.

D.1 Semantic Role Labeling.

We use the questions and sentences provided by the respective datasets. Say, the question and the sentence context were denoted by <ques> and <context> respectively. For the T5 experiments, we follow the QA prompt format as specified by Raffel et al. (2020) which is: "`question: <ques> context: <context>`". For Flan-T5, we use the following prompt design: "`<context> \n In the above sentence, <ques>`". These are the inputs to the respective models. For generation, we considered a beam size of 20. We considered 20 shortest paths to be returned from Yen's K-shortest path algorithm. Generally, a higher K might get better results but we found 20 to be optimal for the perfomance-time tradeoff.

D.1.1 Iterative Prompting Template

In the iterative prompting template, we prompt the model one question at a time such that the answer for every preceding prompt is included in subsequent prompts. Assume a predicate in a sentence to have n questions: $\{q_1,q_2,\ldots,q_n\}$. In the first step, we prompt the model for q_1 using the prompt template mentioned in the previous sub-section. Next, we follow with the prompt the model for q_2 such that the question and the generated answer for q_1 are included in the prompt. Consequently, we use a different prompt template that contains this pair and an instruction that asks the answers to not overlap. Subsequently, the prompt for q_3 will contain the question-answer pair q_1 and q_2, and so on. An example is shown in Figure 11. We use the order provided in the QA-SRL datasets for our experiments.

D.1.2 GPT-4 Prompt Template

We used OpenAI's API and the `ChatCompletion` end-point to obtain the GPT-4 results. We provide the following system instruction: "*You will be given a set of questions regarding a particular sentence. Answer these questions such that the answers to the questions do not overlap. Answers should strictly be a sub-sequence of the sentence. Do not include anything except the answer phrase in the answer*". A user prompt consists of a sentence followed by a set of questions pertaining to a certain predicate in the sentence. As a result, each predicate consists of a request to the endpoint. An example request and answer are shown in Figure 12. A majority of the GPT-4 generations adhere to the answer template shown in the figure. However, for predicates with a single question, answers did not always contain answer headers (e.g., "`Answer 1: <answer>`"). These were handled accordingly. In some rare cases, "`None of the above choices`" was generated and such cases were considered as having no answers. The experiment across the two datasets cost a total of $96.31.

D.2 Coreference Resolution.

We are given the context (<context>) and a pair of mentions(<m1> and <m2>) for the input. For the Macaw models, the input prompt follows this format:

"`$answer$ ; $mcoptions$=(A) Yes (B) No ; <context> Does <m1> refer to <m2>?`"

Prompt:

Elrond gave Aragorn the sword .
In the above sentence, Who gave something?

Flan T5:

Elrond

Prompt:

For the sentence given below answer the following questions such that the answers do not overlap.

Sentence: Elrond gave Aragorn the sword .

Question: Who gave something?
Answer: Elrond

Question: Who was given something? Make sure the answer is present in the sentence.

Flan T5:

Aragorn

Prompt:

For the sentence given below answer the following questions such that the answers do not overlap.

Sentence: Elrond gave Aragorn the sword .

Question: Who gave something?
Answer: Elrond

Question: Who was given something?
Answer: Aragorn

Question: What was given? Make sure the answer is present in the sentence.

Flan T5:

the sword

Figure 11: An example of the iterative prompting template. Note that Flan-T5 is not a stateful model.

System Instruction:

You will be given a set of questions regarding a particular sentence. Answer these questions such that the answers to the questions do not overlap. Answers should strictly be a sub-sequence of the sentence. Do not include anything except the answer phrase in the answer.

User:

Sentence: Elrond gave Aragorn the sword .

Question 1: Who gave something?
Question 2: Who was given something?
Question 3: What was given?

GPT-4:

Answer 1: Elrond
Answer 2: Aragorn
Answer 3: the sword

Figure 12: An example with the template used for GPT-4 experiments.

For the Flan models, we follow the format: "`<context> \n In the above passage, does <m1> refer to <m2>? Yes or No?`"
We restrict generations to {"Yes","No"} for Flan experiments and to {"$answer$ = Yes", "$answer$ = No"} for the Macaw experiments. Integer Linear Programs were implemented and solved using the Gurobi solver (Gurobi Optimization, LLC, 2023).

E Inference Overhead

Since the inference algorithm follows the generation step, a time overhead is expected. Table 10 shows the generation time and its corresponding inference time for the datasets we consider. Crucially, while the generation step is run on the GPU, our inference implementation is run on the CPU.

F SRL Inference Worked-out Example

We will consider the example mentioned in Figure 2 with toy values to illustrate the inference process. In this toy setup, we consider three roles (`a`,`b`, and `c`)[17] and assume the top-2 candidate spans per role with their scores as returned a hypothetical generative model.

The spans and scores for this toy example are given in Table 11. Given these spans and scores. we can construct the directed graph as shown in

[17] You may assume these to be the `Agent`, `Recipient` and `Theme` roles as mentioned in §1

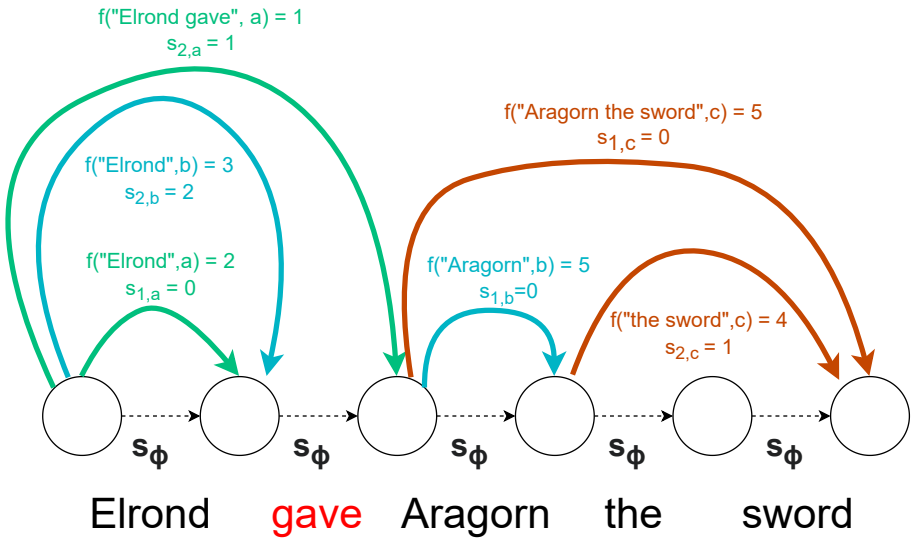

Figure 13: An example of the SRL inference graph with values. Every edge shown in the figure contains the scores returned by the scoring function (in our case, the generative language model) $f(t_r, i)$ where is t_r is the rth ranked span for role i. As mentioned in §4.1, edge score for a span t_r is then calculated as $s_{r,i} = f(t_1, i) - f(t_r, i)$ where $f(t_1, i)$ is the score (as returned by the scoring function) for the top ranked span for role i. Note that we use these scores as the edge weights and, subsequently, as the scores used in the inference algorithm.

Dataset	Gen. step	Inf. Step
QA-SRL$_{wiki}$	48.49	2.18
QA-SRL 2.0	945.22	32.68
ECB+	81.87	0.08
Ontonotes	1547.03	62.99
GENIA	859.88	116.33

Table 10: Time taken (in minutes) by the generation (Gen.) and the inference (Inf.) steps for the datasets. The generation step for QA-SRL$_{wiki}$, QA-SRL 2.0, and Ontonotes were executed on an Nvidia A100 (40 GB) GPU. The generation step for ECB+ and GENIA were executed on an Nvidia A40 (48 GB) GPU. The benchmarking is performed for the T5-3B and Macaw-3B models for the SRL and coreference resolution tasks respectively.

Figure 13. We calculate the edge scores $s_{r,i} = f(t_1, i) - f(t_r, i)$ where $f(t_1, i)$ is the score (as returned by the scoring function) for the top ranked span for role i. For the sake of this illustration, we shall use $s_{r,i}$ as the span notation as well. The s_ϕ value is set to zero. The first step of our inference algorithm aims to find the K shortest paths from the leftmost to the rightmost node. Observe how overlapping spans can never be on the same path. As a results, the top ranked candidates for roles b (span $s_{1,b}$) and c (span $s_{1,c}$) can never be in the same structure.

K shortest paths are returned in increasing order of total path length where path length for a path S is $\sum_{s_{r,i} \in S} s_{r,i}$ as shown in Table 12. As a result, one might notice that partial structures such as paths $\{s_{1,a} \rightarrow s_{1,c}\}$ or $\{s_{1,a} \rightarrow s_{1,b}\}$ might have the shortest path length. However, the QA-SRL framework necessitates one answer per question(or, in other words, role). A post-processing step follows the shortest path step where, from the K shortest paths returned, the shortest path containing exactly one span for each role is chosen. This step also ensures that paths where multiple spans for the same role are present (e.g., $\{s_{2,b} \rightarrow s_{1,b} \rightarrow s_{2,c}\}$), are also avoided. In the example shown in Figure 13, $\{s_{1,a} \rightarrow s_{1,b} \rightarrow s_{2,c}\}$ gives the most optimal path with a path length 1 and satisfies all the constraints. This yields to "*Elrond*", "*Aragorn*" and "*the sword*" as arguments for roles *a*, *b* and *c* respectively. Intuitively, a sufficiently large K (in our case, 20) leads to a higher chance of obtaining a complete and consistent structure. In cases where no complete structures can be formed from any of the K shortest paths, partial (yet, consistent) structures are predicted.[18]

[18]Partial structures can be considered as having "null" answers for some roles. For evaluation purposes, we treat them as incorrect answers.

Rank	Role		
	a	**b**	**c**
1	"*Elrond*" (2)	"*Aragorn*" (5)	"*Aragorn the sword*" (5)
2	"*Elrond gave*" (1)	"*Elrond*" (3)	"*the sword*" (4)

Table 11: Example spans and scores (in brackets) of candidate spans for each role. The scores correspond to the $f(t_r, i)$ values discussed in §4.1 where is t_r is the rth ranked span for role i.

Rank	Path	Path length
1	$\{s_{1,a}, s_{1,b}\}$	0
2	$\{s_{1,a}, s_{1,c}\}$	0
3	$\{s_{1,a}\}$	0
4	$\{s_{1,b}\}$	0
5	$\{s_{1,c}\}$	0
6	ϕ	0
7	$\boldsymbol{\{s_{1,a} \rightarrow s_{1,b} \rightarrow s_{2,c}\}}$	**1**
8	$\{s_{2,a} \rightarrow s_{1,c}\}$	1
	...	

Table 12: K-shortest paths returned by the