Structured Language Generation Model for Robust Structure Prediction

Minho Lee, Junghyun Min, Woochul Lee, Yeonsoo Lee NCSOFT Research Center, Seongnam, Gyeonggi, Korea {minolee, hyun1, darkgeo, yeonsoo}@ncsoft.com

Abstract

Previous work in structured prediction (e.g. NER, information extraction) using single model make use of explicit dataset information, which helps boost in-distribution performance but is orthogonal to robust generalization in real-world situations. To overcome this limitation, we propose the Structured Language Generation Model (SLGM), a framework that reduces sequence-to-sequence problems to classification problems via methodologies in loss calibration and decoding method. Our experimental results show that SLGM is able to maintain performance without explicit dataset information, follow and potentially replace dataset-specific fine-tuning.

1 Introduction

Recent advances in pre-trained language models (PLMs) and large language models (LLMs) allow computational approaches to more tasks and problems than ever. LLMs have demonstrated exceptional end-to-end problem-solving capabilities; however, their internal workings remain largely a black box. Therefore, tasks related to natural language understanding (NLU) still remain significant from the perspective of explainable AI (Ribeiro et al., 2016).

Structure prediction is among many attempts to solve multiple NLU tasks with a single model. Structure prediction is set of natural language understanding tasks, including named entity recognition, relation extraction, and other tasks that requires structure understanding. To perform structure prediction, it is vital that a language model to have a thorough understanding of language and its structure. However, some studies have demonstrated that LLM themselves lack the capability to understand language structure effectively (Zhong et al., 2023; Berglund et al., 2023). We observed that ChatGPT-4¹ (OpenAI, 2024), even when

equipped with detailed instructions and examples, fall short comparing with relatively smaller models, as seen in Table 1. To overcome such shortcomings, some studies like DeepStruct (Wang et al., 2023) and UIE (Lu et al., 2022) proposed pre-training on structured dataset, which contains large number of entities and relations, to improve generative models' understanding ability. DeepStruct used large corpuses like TEKGEN and KELM (Agarwal et al., 2021), which is based on WikiData (Vrandečić and Krötzsch, 2014). UIE used Wikipedia and ConceptNet (Speer et al., 2018) as their pre-training corpus.

Previous works on structure prediction like TANL (Paolini et al., 2021) and Deepstruct reported performance improvements in various structure-related tasks with a single model. However, a nontrivial portion of such improvements represent successful dataset-specific engineering, rather than an overall improvement in structure understanding, as the improved models rely on finetuning and explicit dataset information. Performing inference without explicit dataset information supports this, as we observe a significant drop in benchmark scores and witness a diverse array of formatting errors, as shown in Table 1 and Section 4. We aim to measure and optimize on general structure understanding abilities, rather than task-specific engineering abilities. We also focus on robustness, where the model performs consistently well in a diverse set of situations, including low-resource settings.

In this paper, we propose a new framework named Structured Language Generation Model (SLGM) which can maintain robust structural understanding ability. Our framework comprises of task-specific format information, format loss, and formatted inference. The SLGM framework is illustrated in Figure 1. Task-specific information contains structure of desired output structure, which constitutes the task-specific restrictions along with

¹https://chat.openai.com

Mod	lel	Ent. F1	Rel. F1		
	Original	88.4	72.8		
DeepStruct	- Dataset	81.7	40.7		
	+ Tagset	01.7	70.7		
	- Dataset	79.6	48.2		
TANL	Original	90.3	70.0		
IANL	- Dataset	24.0	2.8		
ChatGl	PT4*	57.7	34.1		

Table 1: F1 Score on CoNLL-04 joint entity and relation dataset of previous works, regarding to the inclusion of dataset or tagset information, and ChatGPT4. We used DeepStruct 10B multi-task model and TANL multi-task model for inference. Score of ChatGPT is measured on first 40 sentence of test dataset. Full prompt we used to inference on ChatGPT4 is shown in Appendix A

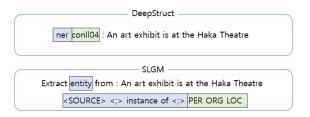


Figure 1: Difference between input structure of previous works and SLGM on NER task. Blue area is task specific information. Green area is dataset specific information. Instead of using dataset name as prompt, we augmented additional format information into model.

possible labels. Format loss works to learn the intended format by penalizing incorrect output format. Finally, formatted decoding encourages the model to decode with format guidelines.

We train and evaluate SLGM on structure prediction tasks. Our experimental results demonstrates significant performance improvements over baseline models of same size which optimizes only with cross entropy loss. Also, SLGM follows performance of models with explicit dataset information and models fine-tuned on specific dataset, which indicates our method can replace dataset-specific engineering.

The contributions of this work are as follows.

- We introduce SLGM, which is able to maintain structural understanding ability well without dataset information. (Section 3)
- We trained and evaluated SLGM on structured prediction task, and observed improvements on almost every task compared to plain cross entropy loss. (Section 4, 5)

• With additional studies, we showed our method can work as zero-weight adapter that simulates dataset specific fine-tuning, and works better on low-resource environments. (Section 6)

2 Related Works

Since emergence of Attention mechanism (Vaswani et al., 2017), many pre-trained language models were introduced to solve NLP tasks. Models like T5 (Raffel et al., 2023), Bart (Lewis et al., 2020), Flan-T5 (Chung et al., 2022), GLM (Du et al., 2022) tried to transfer NLP tasks into sequential generation problem. These models introduced new pre-training methods to solve general tasks, and they showed improvement on zero-shot task scores.

Structure prediction task contains many NLU tasks like NER, RE, SRL. It is part of multitask problem, and requires model to understand structure of languages to achieve high score. Traditional approach, however, created task-specific models for each task. Based on sequence to sequence pretrained LMs, studies like TANL (Paolini et al., 2021), DeepEx (Wang et al., 2021), Deep-Struct (Wang et al., 2023), UIE (Lu et al., 2022), USL (Min et al., 2024) created unified model to solve structure understanding tasks in single model. TANL and UIE created specific output formats for each task. DeepEx and DeepStruct translated every output as triples to generate more unified output across tasks. Additionally, DeepStruct, UIE and USL used structural pre-training method with additional data to improve model's structural understanding ability. Our model adopted some idea from generation tasks to replace dataset specific information or additional training.

In sequence generation tasks, there has been many research on methodologies for integrating words from the original text into generation result. Copying mechanism (Gu et al., 2016) introduced a new method computing the generation score of a token alongside the score of an extracted token from the original text, which enables the direct inclusion of tokens from the original source. This approach was broadly adopted, from summarization (Zeng et al., 2016) to dialogue systems (Park et al., 2023). (Choi et al., 2021) extended copy mechanism, by explicitly decide when to copy and when to generate. By adjusting the weights, it can also be utilized for classification of tokens within a sentence.

Not only using source on generation, there were many attempts using formats or masks when generating sequence. Before machine learning approach were general in NLP area, there were many rule-based approaches (Hovy, 1993). Since the widespread adoption of Pretrained Language Models (PLMs), however, the use of formats or rules for generation has not been extensively researched. On abstract summarization, (Gehrmann et al., 2018) masked some part of source document, not to copy from useless parts. On machine translation (Ghazvininejad et al., 2019) to speed up inference, or sentiment analysis (Li et al., 2020) to create more precise augmentation data. Among generation tasks, code generation (Ling et al., 2016) is one of the most sensitive study related to format. Study like CodePAD (Dong et al., 2023) utilized pushdown automata to generate tokens according to current state. Comparing with code generation, structure prediction has less states and simple, concrete state transition rules, which can be implemented easily.

We thought structured prediction task can be reduced into some point between extractive summarization and classification, with much tighter requirements. We adopted concept from formatted decoding of code generation and copy mechanism of summarization to get higher accuracy and less format misses on structure prediction.

3 Method

Our model operates around a task and data-specific format. We have established a tagset and a specific set of words related to the task and dataset, in order to guide the model about what it should generate. SLGM uses this format to calculate format loss, and formatted decoding.

3.1 Task-specific Format

Tasks like named entity recognition and relation extraction need to extract its element from source sentence. Moreover, within the same task, the tagset can vary depending on the dataset. Previous works inferred valid tagset from dataset names. We introduce additional task-specific format information, which also contains valid tagset, and convey this to model distinct from the input prompt. Task-specific format is used for calculating additional loss and masked inference.

Task-specific format is defined with separator tokens and area between separators, which is called

Slots. A single format string contains multiple separator tokens (<;>) to distinguish slots and a triple separator token (</>) to indicate the end of the format. Each slot can contain a specific string or either two special slot tokens - <ANY> and <SOURCE>. When <ANY> token is given at the slot, it means the model can generate any token. <SOURCE> token means the model should generate tokens only appeared in source sentence. If specific string is given in slot, it means the model should generate one of those tokens given in current slot, which can be used as tagset information. During input generation process, we parse and tokenize the given format string into a format mask tensor. Format mask tensor is a boolean tensor that has true value on legal token ids per each slot. This tensor is used on calculating format losses or masking illegal tokens on decoding process.

3.2 Format Loss

Traditional cross entropy loss used in previous works is calculated over every possible vocabulary defined in model. We thought this is way too broad, so in addition to cross entropy, we introduce structure loss for better format prediction, and slot loss for better tagset prediction.

3.2.1 Structure Loss

Structure loss aims to produce outputs in correct format. We fixed every output into triple format, so it is important to correctly produce separator tokens.

$$L_{st} = \sum_{t \in S} l_t \cdot missed \cdot w_{miss}$$
$$l_t = -log \frac{e^{x_{y_t}}}{\sum_{c \in V} e^{x_t}}$$

where S is location of separator tokens, w_{miss} is miss weight, which is hyperparameter. missed refers to the number of instances where a separator did not have the highest generation probability at its designated position within a sentence. Since wrong generation of separator gives critical parsing issue, structure loss gives bigger penalty of wrong separator prediction.

3.2.2 Slot Loss

Slot loss makes model produce correct tokens among filtered tokens. For example, NER tasks should select tokens from original sentence for head, and specific tags defined in dataset for tail. The formula for calculating slot loss resembles that

of cross entropy; however, the key distinction is in the denominator's token size. Slot loss computation only involves a pre-selected set of legal tokens pertinent to the slot, rather than encompassing all tokens. With slot loss, we can translate sequence generation task into classification task, which has far smaller classes than whole vocabulary size. Slot loss is defined as:

$$L_{sl} = \sum_{k=0}^{N} l_{t,k} (t \in S_k)$$
$$l_{t,k} = -log \frac{e^{x_{y_t}}}{\sum_{c \in V_k} e^{x_{t,c}}}$$

where N is number of slots, S_k is slot k, V_k is legal token set defined at slot k.

Final training loss is calculated as weighted sum of three losses:

$$L = w_{ce}L_{ce} + w_{st}L_{st} + w_{sl}L_{sl}$$

where ce stands for cross entropy, st for structure, and sl for slot. ws are loss weight, which are hyperparameters.

3.3 Formatted Decoding

During inference time, we hold state information of current generation per each sentence. This state information works as simple finite state machine. Each time when the model generates slot separator, state counter increases. When model generates triple separator, state counter sets to 0. Before generating the next token, we retrieve the format mask corresponding to the current generation state of the slot, enforcing the generation of legal tokens by adding a large negative value to the scores of illegal tokens.

4 Experiments

Our research followed the DeepStruct (Wang et al., 2023) experiment protocol, involving two stages: Structural pre-training and multi-task training. DeepStruct also fine-tuned multi-task model with each dataset to get higher result. However, since our main goal is to enhance model performance in multi-task scenarios, we report our main score on multi-task training.

4.1 Tasks and Datasets

4.1.1 Structural Pre-training

We used TEKGEN and KELM (Agarwal et al., 2021) as our pre-training corpus. We extracted entities and relations from given triples, and generated

named entity recognition and relation extraction corpus. Since TEKGEN and KELM only contains surface form of entity but nothing about entity type, we retrieved entity type using WikiData (Vrandečić and Krötzsch, 2014). There are about 50k wikidata entity types and 40k relation types, so instead of using them directly, we mapped frequent entity and relation types into common types. Detailed statistics of mapping result is shown in Appendix B. We used 100k sentences from each corpus, NER and RE respectively (i.e. total 400k sentences).

4.1.2 Multi-task training

For multi-task training, we used dataset shown on Table 2 with given format for multi-task training. Following TANL and DeepStruct, we marked brackets around targets when needed, like head and tail of TACRED relation extraction dataset, or predicates of CoNLL-12 semantic role labeling dataset. We trained 2 epochs using every dataset on structural pretrained model.

4.2 Baseline

We compared our model with various training settings that have same structure, same size, and same data. First, we trained model using cross entropy loss, using same input prompt and answer with SLGM(CE). Also, we trained model using Deep-Struct's prompt (shown in Figure 1), which includes task and dataset information, with cross entropy loss as our oracle model (CE+Data). To see how dataset information affects to multi-task setting, we additionally evaluated DeepStruct prompt model without dataset information(CE+Data*).

4.3 Evaluation Method

We report score and format errors on each dataset. On every task except dialogue state tracking, we evaluated with micro F1 score. For joint entity and relation extraction, we recorded F1 score of entity and relation respectively. For dialogue state tracking task, we used joint accuracy score. We categorized format errors into three types: length mismatch, source mismatch, and tagset mismatch. Length mismatch means length of generated tuple does not match with given format. Source mismatch means the element of output violated <SOURCE> slot token restriction. When the model generated tokens that don't exist in source sentence, source mismatch count increases. Finally, a tagset mismatch occurs when the model generates an output that is not defined within the tags specified by

Task	Dataset	Format
	CoNLL-03 (Sang and De Meulder, 2003)	
	Ontonotes-v5 (Weischedel et al., 2013)	
NER	GENIA (Kim et al., 2003)	<source/> <;> instance of <;> tagset
	CoNLL-04 (Roth and Yih, 2004)	
	NYT (Riedel et al., 2010)	
	TACRED (Zhang et al., 2017)	
RE	TACREV (Alt et al., 2020)	<source/> <;> tagset <;> <source/>
KĽ	CoNLL-04 (Roth and Yih, 2004)	SOURCES <, stagset <, sources
	NYT (Riedel et al., 2010)	
SRL	CoNLL-12 (Pradhan et al., 2012)	<source/> <;> instance of <;> tagset
ID	ATIS (Hemphill et al., 1990)	intent and is an expectation
עו	SNIPS (Coucke et al., 2018)	intent <;> is <;> tagset
DST	MultiWOZ (Budzianowski et al., 2018)	[User] <;> <source/> <;> <any> </any>

Table 2: Task, dataset and task-specific formats. Joint entity and relation extraction corpus like CoNLL 2004 and NYT dataset split one sentence into two sentences, with different prompt and format.

the dataset. We report results averaged on 5 runs.

4.4 Implementation Detail

We used Flan-T5 (Chung et al., 2022), an instruction-tuned version of T5 from HuggingFace hub (Wolf et al., 2020) as our base model. As shown in Figure 1, we used "Extract *target* from: *sentence*" as our input prompt, where *target* is determined by task. We trained our model using base sized checkpoint(220M parameters), with batch size 16 per GPU on 4 NVIDIA A100 GPUs with 40G memory. For SLGM hyperparameters, we used $w_{ce}=0.5, w_{st}=0.2, w_{miss}=0.33$, and $w_{sl}=0.3$. We used greedy decoding when generating answers.

5 Main Result

5.1 Comparing with plain extraction

Table 3 shows our full result. Our model showed better result than models without dataset information, and showed comparative result with dataset-aware model on most dataset. Also, our model generated less format errors than dataset-aware model, which mean our model is much robust on real world dataset. When training without dataset information, we observe an average performance drop of 11 points compared to when dataset information is included. Furthermore, the frequency of format errors increases 12 times larger. Training with dataset information and then removing it yields worse results compared to training without the dataset information from the start. This means model mostly depends on dataset information.

However, it's important to note that CE and CE+Data* operate without any tagset information, whereas SLGM can indirectly acquire tagset information. Since there are multiple datasets for NER and JER, which have their unique tagset, the occurrence of format errors is inevitable. Looking for RE and ID, which share identical or similar tagsets, and for SRL and DST, where only one dataset exists, it is observed that the baseline model also exhibits a low number of format errors. In the following section, we further analyze format errors and experiment to determine whether the use of formatted decoding as a means of providing indirect tagset information is effective for the baseline model as well.

5.2 Comparing with models using formatted decoding

To investigate the impact of formatted decoding on generating the correct format, we applied formatted decoding stage to baseline models and, conversely, removed from our SLGM model before proceeding with inference. Table 4 shows the average score and average format errors across all datasets. Formatted decoding itself can work greatly on extracting correct tagsets. On CE model trained without dataset information, model using formatted decoding achieved 6 point higher F1 score and 7% format errors compared with original model. However, when trained with dataset information, the model cannot recover from absence of dataset information even with format information. F1 score and format errors are improved, but still showed large number

Task	Dataset	(CE	СЕн	-Data*	SLO	GM	CE+	Data
Task	Dataset	Score	FE	Score	FE	Score	FE	Score	FE
	CoNLL-03	69.32	4700	12.06	10273	80.28	43	87.11	5
NER	OntoNotes	75.44	766	24.79	5143	75.87	142	81.12	348
	GENIA	67.05	147	65.26	70	69.88	5	66.48	30
RE	TACREV	66.95	6	66.52	21	71.39	2	67.04	13
KE	TACRED	58.41	6	59.51	21	63.11	2	59.07	13
	CoNLL-04	0.00	873	0.00	827	71.74	0	74.88	14
JER	CONLL-04	13.31	413	23.08	491	27.87	2	48.53	46
JEK	NYT	88.68	322	74.60	412	88.80	149	88.45	23
	NII	65.80	17	45.07	464	59.36	20	67.47	7
SRL	CoNLL-12	82.47	161	82.37	158	83.45	0	82.35	159
ID	ATIS	94.09	11	94.30	15	93.96	0	94.21	9
ш	SNIPS	96.58	0	95.79	1	96.86	0	96.07	1
DST	MultiWOZ	38.16	667	36.68	0	38.87	0	37.18	0
A	verage	62.79	622.23	52.31	1376.62	70.88	28.08	73.07	51.38

Table 3: Main result of our experiment on multi-task setting. FE stands for "Format Error". Scores are higher the better, and format errors are lower the better. Average scores and format errors are dataset-wise macro average.

Loss	FD	Avg. F1	Avg. FE
CE		63.03	625.2
CE	О	69.87	44.78
CE+Data*		52.73	1404.98
CE+Data ·	О	59.32	568.91
CE+FL		63.51	581.62
CLTIL	О	70.86	28.52

Table 4: Average score regarding to loss and inference method. FD stands for Formatted Decoding. FL on last two rows stands for Format Loss. Note that model trained with cross entropy and format loss, using formatted decoding(CE+FL+FD) is equal to SLGM.

of format errors comparing with other models using formatted decoding. When designing models with real-world applications in mind, it becomes evident that including dataset names during training can lead to undesirable effects.

When comparing models without formatted decoding, model with format loss showed best score and least format errors. We think this means format loss itself, works as dataset indicator. Slot losses produces tagset-specific loss, and this applies stronger back-propagation than cross entropy, which applies on every tokens. This makes model know more about what to extract on given sentence, which indicates the model understands structure of sentence better.

Full result on each dataset is shown on Appendix D.1

5.3 Format Error Analysis

We analyzed ratio of format errors for each model, with and without formatted decoding. Figure 2 shows format error count and error type ratio across tasks. As illustrated in the figure, the absence of dataset information most prominently results in tagset mismatches. Formatted decoding can reduce tagset mismatch dramatically. It could not completely remove them, since formatted decoding only masks on tokens, not sequences. In cases where tagset mismatches were not resolved, the majority involved generating shorter tags with the same meaning, such as producing only "LOC" when "LOCATION" was required. For reference model trained with dataset information, source mismatch were more frequent than other models. Since it is tuned with dataset information, the model can clearly distinguish legal tagset. However, it cannot correctly infer correct source tokens, especially in named entity recognition.

6 Additional Studies

We conducted additional studies to see in what situation our model works well. In this section, we report average result across all datasets with single run. Full result is shown on Appendix D.

6.1 Fine-tuning

We sought if our model is competitive enough with fine-tuning. Since fine-tuning involves train-

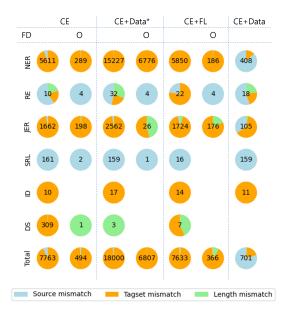


Figure 2: Total format error counts and ratio per tasks. Empty area means there are no format errors in such setting. Number in circle means sum of format errors across every dataset inside given task.

Loss	FT	Avg. F1	Avg. FE
CE		53.10	742.31
CE	О	77.70	56.45
CE+Data*		53.15	1403.53
CE+Data.	О	63.53	606.00
CE+FL		66.74	503.76
CE+FL	О	78.93	16.09
SLGM		73.37	18.69
SLOW	О	78.85	0.64

Table 5: Average score and format error regarding to fine-tuning. FT stands for fine-tuning. SLGM equals to CE + Format loss + Formatted decoding.

ing on a single dataset with clean tagset, it can be expected to improve performance even for models without dataset information. Based on models trained in multi-task environment, we conducted an additional 5 epochs of training for each dataset and then evaluated their performance. Table 5 shows the result. Comparing with result of Table 4, finetuning can achieve relatively higher F1 score than formatted decoding. Furthermore, when looking at fine-tuning result of CE+FL and SLGM, we can observe that formatted decoding has minimal impact on performance. These results lead us to believe that formatted decoding can partially replicate the effects provided by fine-tuning. This can be seen as similar to the principle of operation of adapters (Houlsby et al., 2019), in that the original model

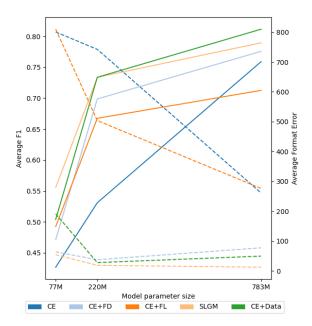


Figure 3: Average score and format error according to model size(small-base-large). Solid line indicates Average F1 score(left side). Dotted line indicates format errors(right side).

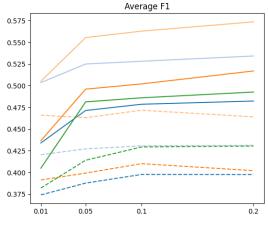
remains unchanged, and additional lightweight features are incorporated.

6.2 Model size

We additionally evaluated our loss function and formatted inference on Flan-T5-small(77M) model and Flan-T5-large(783M) model. Train and inference setting is same as written in Section 4. Summary of results is shown in Figure 3.

Our model also showed better result than CE with dataset information on small sized model. Since smaller models that doesn't have enough language capacity are sensitive on additional information, our format loss and formatted inference provides stronger hint on correct labels than plain cross entropy loss. On large sized model, plain cross entropy model got much better result than smaller models. With more model power, it can recognize characteristics of each dataset without any hints. Giving indirect hint with format loss, however, showed negative effect on large size. Still, with formatted decoding, model with format loss performed better than cross entropy model. We can infer format loss and formatted decoding should mixed together to get better results.

When using our best methods, we achieved state of the art performance in some datasets (Table 12).



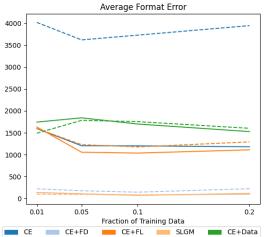


Figure 4: Average score and format errors according to dataset fraction. Solid line represents model with structural pre-training, and dotted line represents model without structural pre-training.

We achieved 94.8 micro F1 score on CoNLL-03 NER dataset, and 98.3 micro F1 score on ATIS intent detection dataset, which are both state of the art. This indicates that our method can further improve through fine-tuning, and it can achieve better performance without the need for engineering specific to dataset features.

6.3 Low-resource setting

We hypothesized that SLGM effectively retains the characteristics of data, prompting us to investigate its performance in low-resource environments through experimentation. For each multitask dataset, we extracted data with proportional tag distributions, with at least one example, to use as training data. We didn't used MultiWOZ dataset since it is hard to distinguish unique tags. To compare true power of our model, we trained in two different environment, with and without structural

pre-training.

Figure 4 shows our experiment result. Regardless of structural pre-training, SLGM worked better than other models. SLGM even showed higher score and lower format errors than dataset aware model on low-resource environment. We also observed that formatted decoding is better to use than format loss on low-resource setting. Additionally, it is notable that models using format loss showed worse score when trained without structural pretraining and have more data.(CE+FL, SLGM) In our observation, the model showed stable scores across most dataset, but showed steep increase of hallucinations in NYT relation extraction task. We speculate that the use of format loss in low-resource environments makes it challenging to accurately learn the proper termination point, resulting in repetitive outputs.

Comparing dotted line with solid lines, structural pre-training has great improvements on all training settings. With structural pre-training, models did not exhibit a decline in performance regardless of dataset fraction. This shows structural pre-training enhances the stability of the model and can be particularly helpful in guiding how the model interprets sentences, especially when data is scarce.

7 Conclusion and Discussion

We introduced a new framework named SLGM, which utilizes task-specific formats. By using formats with format loss and formatted decoding, we reduced structure prediction task into classification tasks. We showed our models were more robust on multi-task environments without dataset information, which is more powerful on real-world scenario. Our method worked similarly as using fine-tuning on each dataset. With using bigger model and fine-tuning, our model still works better and achieved state of the art on some datasets.

We successfully replaced dataset information into formats, which can be expanded in various ways. Our research has only applied the fundamental concepts, and we think formats and formatted decoding can be more generalized. For example, we can remove unnecessary fixed slots, or create more complex formats with expanding decoding states using finite state machine. Our model has possibility on those situations too. We anticipate that format loss and formatted decoding will be effectively utilized in fields beyond structured prediction.

8 Limitations

Format loss and formatted decoding represent basic strategies designed to compel the model to produce outputs in accordance with a predetermined format. There may be many improvements on both scoring and real-world usage. There may be many problems in specific situations. Suppose we extract named entity with location but there is no location tag in format. The model expected location and extracted some words, but the location tag is illegal. In plain decoding strategy, the model would generate any token similar to location. However, when using formatted decoding, it would generate unexpected string, which could be worse than extracting wrong tag. We did not conduct quantitative experiments regarding this situation.

This scenario could potentially be addressed by employing a beam decoding strategy, which operates on triple units. This issue is related to the fact that the model does not know about format during actual attention calculation. Despite explicit format is given, model cannot utilize format information at attention layers. We think additional cross attention layer over formats may show better results. We leave these possible improvements as future works.

9 Ethics Statement

In our research, we utilized a pretrained model and made use of publicly available datasets published on the web. We acknowledge that the data obtained from the web may contain potential biases. Our model can bear potential risks and harms discussed in (Brown et al., 2020), and we clarify that our research did not specifically address or consider these risks. We ensured that all datasets employed in our study were accessed and used in a manner that respects their intended use and complies with any associated licenses or terms of service. We are also mindful of the potential biases present in these datasets and the pretrained model.

We used ChatGPT as a tool for generating and refining text content. We acknowledge and address the ethical considerations associated with the use of such AI technology, and have thoroughly reviewed the content to ensure that it does not include any unethical material.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic

corpus generation for knowledge-enhanced language model pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3554–3565, Online. Association for Computational Linguistics.

Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. 2020. Tacred revisited: A thorough evaluation of the tacred relation extraction task.

Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2023. The reversal curse: Llms trained on "a is b" fail to learn "b is a".

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.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Sanghyuk Choi, Jeong in Hwang, Hyungjong Noh, and Yeonsoo Lee. 2021. May the force be with your copy mechanism: Enhanced supervised-copy method for natural language generation.

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.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv* preprint arXiv:1805.10190.

Yihong Dong, Xue Jiang, Yuchen Liu, Ge Li, and Zhi Jin. 2023. Codepad: Sequence-based code generation with pushdown automaton.

- Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling.
- Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792*.
- Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models.
- Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.
- Charles T Hemphill, John J Godfrey, and George R Doddington. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language:* Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.
- Eduard H Hovy. 1993. Automated discourse generation using discourse structure relations. *Artificial intelligence*, 63(1-2):341–385.
- J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.
- Kun Li, Chengbo Chen, Xiaojun Quan, Qing Ling, and Yan Song. 2020. Conditional augmentation for aspect term extraction via masked sequence-to-sequence generation.
- Wang Ling, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, Andrew Senior, Fumin Wang, and Phil Blunsom. 2016. Latent predictor networks for code generation.
- Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. *arXiv preprint arXiv:2203.12277*.
- Junghyun Min, Minho Lee, Woochul Lee, and Yeonsoo Lee. 2024. Punctuation restoration improves structure understanding without supervision.

- OpenAI. 2024. ChatGPT. Available online.
- Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages.
- Cheonyoung Park, Eunji Ha, Yewon Jeong, Chi-young Kim, Haeun Yu, and Joo-won Sung. 2023. CopyT5: Copy mechanism and post-trained t5 for speechaware dialogue state tracking system. In *Proceedings of The Eleventh Dialog System Technology Challenge*, pages 89–94, Prague, Czech Republic. 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.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III 21*, pages 148–163. Springer.
- 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.
- Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv* preprint cs/0306050.
- Robyn Speer, Joshua Chin, and Catherine Havasi. 2018. Conceptnet 5.5: An open multilingual graph of general knowledge.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2021. Zero-shot information extraction as a unified text-to-triple translation.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2023. Deepstruct: Pretraining of language models for structure prediction.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. Ontonotes.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. 2016. Efficient summarization with readagain and copy mechanism.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Conference on Empirical Methods in Natural Language Processing*.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2023. Can chatgpt understand too? a comparative study on chatgpt and fine-tuned bert.

A Instruction of ChatGPT4

The result of Table 1 is measured with custom ChatGPT 4. We used custom prompt which defines task and tagset, with two examples to show exact format. Full prompt is shown below:

You are given multiple sentences. You have to extract relations between entities in given sentences. For each sentence, extract every named entity that have relations first. Types of named entity must be one of "location", "organization", "other", "human". Then, extract relations between extracted named entities. Types of relation must be one of "kills", "lives in", "works for", "located in", and "organization based in".

You must FOLLOW given structure. When extracting named entities, you must extract in this form: [ENTITY] <;> instance of <;> [TYPE] . When there are multiple named entitis. separate with </> . When extracting relations, you must extract in this form: [HEAD] <;> [TYPE] <;> [TAIL] . Same with named entity extraction, if there are multiple relations, separate with </> .

For each sentence, first line should be result of named entity extraction, and second line should be result of relation extraction. When given multiple sentence(separated by empty line), pad empty line between extractions.

When given a file, each line contains single sentence.

Example

Input

John Wilkes Booth , who assassinated President Lincoln , was an actor .

The opera company performed at the Palace of Fine Arts , in San Francisco , on June 30 and July 1-2 , said Kevin O 'Brien , a spokesman for the theater

<input end>

Output

John Wilkes Booth <;> instance of <;> human </>President Lincoln <;> instance of <;> human

John Wilkes Booth <;> kills <;> President Lincoln

Palace of Fine Arts <;> instance of <;> location </> San Francisco <;> instance of <;> location </> June 30 <;> instance of <;> other </> July 1-2 , <;> instance of <;> other </> Kevin O 'Brien <;> instance of <;> human

Palace of Fine Arts <;> located in <;> San Francisco

<output end>

With given prompt, we uploaded test file and got result of first 40 sentences.

B Type Mapping of Pre-training Data

We collected entity types appeared more than 20k times, and manually labeled them into 5 different entity types: Person, Location, Organization, Product, and Event. Surface forms that we could not find from WikiData are not used. Table 6 shows example and part of statistics of entity type mapping. Table 7 contains full statistics of mapped entity types.

Similarly, we collected relations containing entities that have valid types mapped at Table 6. Among these relations, we filtered out relations appeared less than 10k times. There were 168 relation phrases meeting these conditions. We additionally clustered 82 of 168 relation phrases into 35 similar clusters. Table 8 shows statistics of mapped relations.

C Dataset Statistics

Table 9 shows statistics of multi-task dataset. Note that TACRED and TACREV shares same number of sentences; their difference is on validation and test labels. On training with format loss, we removed sentences which have labels violating format errors (less than 1%). On evaluation, we used every sentences as-is.

Original	Count	Mapping	Example
type			
Human	9.7M	Person	Umberto
			II of Italy
Country	4.2M	Location	England
Taxon	1.2M	-	Natuna
			Island
			surili
association	954k	Organizaiton	FC
football			Baden
club			
Film	515k	Product	Avengers
Summer	185k	Event	2024
Olympic			Summer
Games			Olympics

Table 6: Entity type mapping examples

Type	Count
Location	10,594,011
Person	10,239,553
Organization	3,182,685
Product	2,233,553
Term	1,186,434
Event	514,083

Table 7: Entity type mapping statistics

Count
3,233,250
1,768,459
1,657,574
1,034,479
1,009,859
937,797
879,955
737,735
496,122
446,835
353,896
300,792
299,818
253,338
162,945
146,352
142,878
122,463
110,483
92,805
84,473
81,600
77,977
76,344
76,093
69,833
61,688
60,142
49,232
42,186
39,123
38,671
31,669
20,402
15,840

Table 8: Relation type mapping

D Detailed experiment result

This section shows full result of Section 5 and 6. We report scores and format errors on every dataset for each settings.

D.1 Formatted decoding

Table 10 shows full result using formatted decoding (Section 5.2), averaged on 5 runs. Comparing with Table 3, formatted decoding greatly reduces format errors on NER and JER tasks. It also shows great score improvements on low resource data

Task	Dataset	Train	Dev	Test
	CoNLL-03	14986	3465	7148
NER	Ontonotes-v5	75187	9603	9479
	GENIA	15023	1669	1854
JER	CoNLL-04	922	231	288
JEK	NYT	56196	5000	5000
RE	TACRED*	68124	22631	15509
SRL	CoNLL-12	253070	35297	24462
ID	ATIS	4478	500	893
שנו	SNIPS	13084	700	700
DST	MultiWOZ	62367	7371	7368

Table 9: Statistics of each multi-task dataset. Each cell represents sentence number.

like CoNLL-04. Even if the model knows about dataset information, formatted decoding still upgrades score and reduces format errors.

D.2 Fine-tuning

Table 11 shows full result of fine-tuning (Section 6.1) on single run. Comparing with Table 10, finetuning shows 5 to 7 points better F1 score than formatted decoding. Surprisingly, best setting utilizing fine-tuning is using only format loss, which is better than using both formatted decoding. Model with format loss win over model with dataset information on almost every dataset. This is evidence that our format loss can reduce problem as easy as classification. However, using both fine-tuning and formatted decoding seems making conflict to each other. This is because some gold instances violates format restriction (e.g. <SOURCE> restriction in named entity recognition). Without formatted decoding, model can infer tokens that did not appeared in source sentence, when training example exists. However, we think this is not good for realworld usage, because it can be seen as overfitting on erroneous training examples.

D.3 Model size

Table 12 shows full result regarding to model size (Section 6.2) on single run. On cross entropy model, model can distinguish characteristics of dataset when model parameters increase. The power of formatted inference on score is reduced when parameter size increased, yet still effective reducing format errors. Comparing CE+FL with CE, using only format loss showed worse score and more format error on large sized model. However, when mixed with formatted decoding, it showed additional increase comparing with models using

single methods. Additionally, when we use every method we tried in this paper (i.e. format loss, formatted decoding, using bigger model, fine-tuning on dataset), we achieved state of the art performance on CoNLL-03 named entity recognition task and ATIS intent detection task. This means our framework still has room for improvement.

D.4 Low resource

Table 13 and 14 shows full result of low resource settings, without and with structural pre-training on single run. As mentioned earlier, when looking at result of CE+FL and SLGM with NYT dataset of Table 13, there were weird explosion of format errors on some dataset fractions, which caused decreased score. We think this happened because without structural pretraining, model does not have enough structural understanding ability. With given small examples, the model tried to extract same entities multiple time when sentence goes longer.

Task	Dataset	(Œ	CE-	+FD	CE	+FL	SLO	GM	CE+Da	ata*+FD	CE+Da	ata+FD
Task	Dataset	Score	FE	Score	FE	Score	FE	Score	FE	Score	FE	Score	FE
	CoNLL-03	69.32	4700	78.09	79	66.17	5127	80.28	43	28.30	2196	87.11	0
NER	OntoNotes	75.44	766	76.04	288	75.50	572	75.87	142	26.72	5164	81.64	45
	GENIA	67.05	147	66.31	9	70.31	75	69.88	5	64.95	3	66.35	1
RE	TACREV	66.95	6	66.93	2	71.51	12	71.39	2	66.48	2	67.00	2
KE	TACRED	58.41	6	58.42	2	63.21	12	63.11	2	59.48	2	59.05	2
	CoNLL-04	0.00	873	66.73	0	0.00	873	71.74	0	66.02	0	74.50	0
JER		13.31	413	26.49	3	16.84	493	27.87	2	27.23	2	45.71	2
JEK	NYT	88.68	322	88.45	156	88.84	300	88.80	149	73.79	17	88.36	1
	NII	65.80	17	65.75	0	59.42	33	59.36	20	45.03	8	67.47	2
SRL	CoNLL-12	82.47	161	82.35	3	83.44	19	83.45	0	82.27	1	82.24	1
ID	ATIS	94.09	11	93.96	0	94.14	12	93.96	0	93.85	0	94.07	0
110	SNIPS	96.58	0	96.58	0	96.86	0	96.86	0	95.72	0	96.01	0
DST	MultiWOZ	38.16	667	38.21	0	38.85	7	38.87	0	36.68	0	37.14	0
A	verage	62.79	622.23	69.56	41.69	63.47	579.62	70.88	28.08	58.96	568.85	8.85 72.82 4	

Table 10: Full result of F1 scores and format errors regarding to formatted decoding.

Task	Dataset	CE-	+FT	CE+Fl	D+FT	CE+F	L+FT	SLGN	1+FT	CE+Da	ata*+FT	CE+Da	ata+FT
lask	Dataset	Score	FE	Score	FE	Score	FE	Score	FE	Score	FE	Score	FE
	CoNLL-03	91.63	10	92.29	0	93.65	4	93.65	0	77.99	2733	91.65	7
NER	OntoNotes	82.26	376	82.23	4	84.22	11	84.21	0	41.93	2860	82.29	365
	GENIA	75.78	5	68.09	0	75.44	6	75.42	0	66.72	18	75.70	4
RE	TACREV	75.18	8	74.57	3	76.95	6	76.90	2	73.12	7	74.58	10
KE	TACRED	65.78	8	64.94	3	66.87	6	66.82	2	64.96	7	65.51	10
	CoNLL-04	83.50	2	82.68	0	85.60	3	85.48	0	0.00	877	84.86	1
JER		52.43	6	50.85	0	55.71	14	55.24	0	36.35	126	50.36	4
JEK	NYT	90.11	11	90.08	0	90.66	13	90.68	0	85.28	15	90.36	20
	NII	74.70	35	74.47	0	76.78	102	76.46	3	62.83	93	75.15	11
SRL	CoNLL-12	82.75	158	82.11	1	84.44	11	84.45	0	82.21	17	82.99	163
ID	ATIS	97.98	2	98.09	0	98.04	1	97.98	0	97.70	1	97.48	1
ID	SNIPS	98.43	0	98.43	0	96.72	0	96.72	0	98.00	0	98.15	0
DST	MultiWOZ	39.61	0	39.35	0	41.02	0	41.02	0	38.85	518	39.86	4
A	verage	77.70	47.77	76.78	0.85	78.93	13.62	78.85	0.54	63.53	559.38	77.61	46.15

Table 11: Full result of F1 scores and format errors regarding to dataset specific fine-tuning.

Task	Dataset		CE			CE+FD			CE+FL			SLGM		C	E+Data	1	S	LGM+F	T
lask	Dataset	S	В	L	S	В	L	S	В	L	S	В	L	S	В	L	S	В	L
	CoNLL-03	48.13	64.03	85.69	58.69	80.36	86.37	54.70	72.76	83.75	70.04	84.18	89.72	71.69	89.79	92.72	88.17	93.65	94.80
	CONLL-03	5684	5843	1756	9	43	5	6108	4437	2531	36	25	18	374	15	45	0	0	0
NER	OntoNotes	48.80	71.89	84.46	49.46	78.20	86.36	61.89	81.30	85.02	61.73	81.64	85.33	66.95	83.84	87.28	76.87	84.21	87.89
NEK	Ontonotes	2080	717	555	360	191	42	1236	411	291	139	99	81	98	83	375	1	0	0
	GENIA	45.97	0.00	77.13	41.75	66.19	72.48	50.74	72.37	76.00	50.22	72.24	75.89	46.18	67.37	76.99	64.08	75.42	76.99
	GENIA	193	1137	5	8	15	7	197	31	16	12	7	4	92	35	0	1	0	0
	TACREV	12.05	54.37	79.24	9.30	65.21	77.74	41.47	72.97	79.34	41.49	72.93	79.35	13.15	66.72	78.57	62.22	76.90	79.10
RE	IACKEV	7	5	9	8	2	2	56	7	9	4	2	4	4	7	14	4	2	2
KE	TACRED	10.72	49.18	69.34	8.24	56.92	67.53	36.81	64.35	69.23	36.80	64.32	69.25	10.86	57.70	68.73	55.35	66.82	69.24
	TACKED	7	5	9	8	2	2	56	7	9	4	2	4	7	7	15	4	2	2
		0.00	0.00	40.31	49.30	67.57	75.22	0.00	6.88	40.74	57.55	71.17	77.39	48.24	74.43	85.56	77.16	85.48	88.18
	CoNLL-04	690	794	643	0	0	0	702	874	635	0	0	0	420	11	1	0	0	0
	CONLL-04	0.00	4.87	45.77	7.98	27.28	45.07	0.00	23.32	38.61	9.32	33.90	44.84	0.00	46.43	59.69	26.00	55.24	60.02
JER		470	530	151	91	0	7	883	425	208	244	2	2	981	99	19	1	0	1
JEK		80.06	85.35	93.59	79.68	88.26	88.36	81.24	89.79	93.66	81.25	89.80	93.69	82.78	88.83	93.76	85.10	90.68	93.85
	NYT	564	475	93	322	232	939	446	215	77	200	95	47	16	37	0	0	0	0
	NII	45.68	54.79	84.66	45.20	66.74	84.98	29.99	65.01	85.22	29.63	64.75	85.21	49.72	66.01	86.06	38.95	76.46	86.66
		298	82	3	5	1	2	600	120	8	59	11	1	68	38	5	45	3	0
SRL	CoNLL-12	65.54	80.22	87.30	65.07	82.51	86.93	72.80	84.72	87.67	72.81	84.72	87.67	65.95	82.15	87.42	73.72	84.45	87.12
SKL	CONLL-12	167	20	161	2	1	0	83	14	9	4	0	1	164	16	162	4	0	1
	ATIS	79.81	92.64	97.87	81.38	93.72	97.87	85.25	96.51	97.92	85.78	96.52	97.76	83.92	94.36	97.92	96.19	97.98	98.32
ID	AHS	134	31	4	17	0	0	123	8	3	5	0	0	96	10	5	0	0	0
ш	SNIPS	89.68	95.22	97.50	89.87	96.43	97.43	92.86	96.72	97.43	93.01	96.72	97.43	88.24	96.86	97.43	96.29	96.72	98.15
	SIMIS	6	1	1	0	0	0	2	0	0	0	0	0	8	0	0	0	0	0
DST	MultiWOZ	28.42	37.75	43.72	26.88	38.89	42.00	32.53	40.96	42.72	32.58	40.96	42.72	27.56	39.33	42.69	35.08	41.02	42.23
DSI	MultiWOZ	106	10	4	0	0	0	36	0	0	0	0	0	155	4	4	0	0	4
	vouces	42.68	53.10	75.89	47.14	69.87	77.57	49.25	66.74	75.18	55.56	73.37	78.94	50.40	73.37	81.14	67.32	78.85	81.73
L_A	werage	800.46	742.31	261.08	63.85	37.46	77.38	809.85	503.77	292.00	54.38	18.69	12.46	191.00	27.85	49.62	4.62	0.54	0.77

Table 12: Full result regarding to model size. S, B, L stands for small, base, large, respectively. For each dataset, upper row is F1 score, and lower row is format errors. Text with bold means state of the art performance.

Task	Dataset	CE				CE+FD					CE+FL				SLGM				CE+Data			
Task		0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	
	CoNLL-03	26.19	42.80	42.71	45.30	51.65	55.67	54.10	55.24	24.72	41.60	44.79	44.32	55.63	60.61	59.42	61.32	26.19	44.20	45.41	45.42	
		10169	7812	7820	7853	2	33	32	30	9937	7516	7820	7822	14	12	15	20	10169	9828	9686	8658	
NER	OntoNotes	32.09	36.05	36.42	37.78	29.55	34.76	35.25	35.66	39.97	40.62	40.25	44.26	36.00	38.80	38.78	41.39	32.09	42.05	39.90	43.55	
HISK		3599	3442	3452	3392	480	752	612	586	4705	3628	3402	3472	610	569	592	405	3599	7886	8326	7794	
	GENIA	42.69	45.87	45.62	42.71	38.07	39.58	39.54	38.24	45.61	44.59	46.95	46.48	44.54	43.99	45.86	46.01	42.69	48.26	47.79	47.00	
		379	319	236	217	82	51	43	89	399	290	301	236	57	48	27	68	379	858	879	586	
	TACREV	0.06	0.00	0.00	0.00	1.08	2.98	0.89	1.69	1.70	6.99	1.52	3.02	1.76	7.04	1.52	3.02	11.49	4.70	12.18	9.75	
RE		30361	29257	30610	33136	1144	914	623	1264	39	17	15	20	4	2	3	3	80	12	19	25	
KE	TACRED	7.87	5.39	10.67	7.10	3.68	2.78	1.31	0.42	1.48	6.07	1.43	2.78	1.53	6.12	1.43	2.78	5.91	3.57	10.88	7.20	
		75	12	16	16	3	3	3	2	39	17	15	20	4	2	3	3	16	33	49	16	
	CoNLL-04	0.00	0.00	0.00	0.00	39.00	41.08	41.64	41.35	0.00	0.00	0.00	0.00	56.38	53.78	53.77	56.11	0.00	0.00	0.00	0.00	
		762	714	716	740	0	0	1	1	782	702	711	704	1	15	2	0	762	632	646	645	
		0.00	0.00	0.00	0.00	0.44	1.34	1.92	1.85	0.00	0.00	0.00	0.00	4.31	3.92	5.00	2.52	0.00	0.00	0.00	0.00	
JER		464	701	691	665	151	42	27	30	446	846	602	943	19	16	19	29	464	611	506	554	
JEE	NYT	77.64	80.34	80.66	78.57	77.22	80.24	80.32	78.35	79.33	81.23	80.92	80.56	79.21	81.17	80.64	80.40	77.64	82.85	82.66	82.03	
		1143	341	459	788	739	244	363	622	1101	348	493	541	440	224	310	369	1143	252	363	546	
		48.18	31.60	33.65	30.24	47.83	31.15	32.04	28.94	52.19	25.60	41.50	20.19	51.94	25.46	41.13	19.76	48.18	37.27	39.00	38.39	
		134	280	354	292	6	60	22	42	154	572	407	1482	1	129	17	293	134	188	127	143	
SRL	CoNLL-12	49.40	56.52	60.59	65.31	48.36	56.17	60.26	64.83	54.16	60.45	64.43	68.33	53.68	60.35	64.48	68.34	49.40	60.39	63.50	68.33	
OIL.	COLUEN 12	960	413	232	121	30	12	3	4	1393	680	190	120	11	78	3	3	960	1005	350	159	
	ATIS	79.54	80.07	79.45	80.64	81.20	79.48	80.99	80.59	82.70	82.68	81.39	82.29	85.18	84.09	84.22	84.09	79.54	85.85	85.66	84.77	
ID		121	117	129	109	13	10	11	12	142	115	145	124	9	12	10	5	121	92	93	94	
	SNIPS	85.37	86.69	87.56	89.57	86.59	87.45	88.59	90.30	87.84	89.37	89.02	90.34	89.02	90.44	89.87	91.16	85.37	87.80	88.33	90.17	
		57	27	27	22	0	0	0	0	47	28	27	25	0	0	0	0	57	8	14	9	
Δ	verage	37.42	38.78	39.78	39.77	42.05	42.72	43.07	43.12	39.14	39.93	41.02	40.21	46.60	46.31	47.18	46.41	38.21	41.41	42.94	43.05	
Arctage		4018.67	3619.58	3728.50	3945.92	220.83	176.75	145.00	223.50	1598.67	1229.92	1177.33	1292.42	97.50	92.25	83.42	99.83	1490.33	1783.75	1754.83	1602.42	

Table 13: Full result of low resource experiment without structural pre-training. For each dataset, upper row is F1 score, and lower row is format errors.

Task	Dataset	CE				CE+FD				CE+FL				SLGM				CE+Data			
lask	Dataset	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2	0.01	0.05	0.1	0.2
NER	CoNLL-03	25.65	44.89	43.35	44.44	51.74	57.90	57.62	57.52	25.80	45.63	45.04	47.94	51.04	62.62	61.09	59.88	23.70	49.10	47.50	52.50
		10480	8083	7859	7781	8	25	21	26	10184	7603	7353	7420	32	28	29	27	10380	9046	8494	7328
	OntoNotes	36.26	41.34	42.19	44.24	36.33	41.39	41.35	43.11	39.09	44.56	44.52	48.91	38.13	45.45	44.35	46.46	35.89	45.29	44.14	46.36
		3108	2397	2641	2741	369	664	452	539	3884	2200	2510	3069	594	712	408	733	4272	8931	8713	8221
	GENIA	45.23	51.49	53.09	48.07	41.38	45.92	47.78	45.43	47.84	53.07	55.02	54.70	43.82	47.42	49.64	49.04	46.36	51.73	54.35	52.91
		528	347	335	300	179	118	88	86	552	372	277	230	200	122	97	135	594	841	748	476
	TACREV	33.68	39.19	43.60	42.50	31.84	35.56	34.53	33.29	20.42	40.41	40.98	44.76	20.99	40.30	41.20	44.86	5.56	30.49	34.92	34.28
RE	IACKLY	94	33	91	59	4	3	3	3	118	68	71	100	2	2	2	2	37	22	53	35
	TACRED	38.95	44.61	45.96	44.77	38.59	44.51	46.03	44.24	39.56	45.99	46.20	45.84	38.82	45.92	46.26	46.38	28.07	30.90	25.74	21.07
		597	579	568	588	3	12	12	3	622	73	31	83	5	2	3	3	202	64	36	35
	CoNLL-04	0.00	0.00	0.00	0.00	59.66	55.38	55.57	57.02	0.00	0.00	0.00	0.00	61.46	57.19	58.31	58.93	0.00	0.00	0.00	0.00
		758	758	702	756	0	2	1	1	741	710	731	789	0	0	1	0	749	688	664	654
		0.47	0.00	0.00	0.00	4.45	7.03	6.47	8.11	0.00	0.00	0.00	0.00	3.72	5.12	6.64	7.79	0.00	0.00	0.00	0.00
JER		656	1037	993	693	208	152	12	12	686	648	593	547	275	54	78	23	463	524	425	511
JEE	NYT	78.43	81.58	81.09	79.99	78.19	81.12	81.05	79.72	79.90	83.02	82.95	82.15	79.53	83.09	82.86	81.93	79.35	82.34	82.33	82.07
		1214	417	421	754	744	158	241	442	1036	326	338	699	461	301	172	470	913	621	565	627
		41.44	26.18	26.54	31.34	40.03	26.11	26.01	30.43	43.96	42.66	44.27	48.97	42.64	41.54	42.81	47.67	47.58	47.69	52.20	52.71
		387	347	470	262	61	114	99	60	458	242	216	180	72	40	25	14	133	185	144	151
SRL	CoNLL-12	52.57	61.29	64.54	68.26	51.64	61.03	64.41	67.93	55.36	63.31	66.76	70.22	54.29	62.98	66.61	69.95	52.49	61.58	66.04	69.87
		1140	353	230	137	6	5	5	1	1182	344	229	141	5	4	2	1	3008	1082	467	214
	ATIS	82.21	84.14	83.07	83.76	83.82	83.63	82.43	82.99	83.40	84.56	84.62	85.13	83.18	83.00	83.66	83.66	80.95	86.82	84.73	87.15
ID	SNIPS	124	98	105	95	35	19	20	20	107	85	91	73	24	13	15	15	145	81	79	76
		85.97	91.03	90.95	91.45	86.59	90.44	90.58	91.30	88.62	92.04	92.13	91.78	88.59	91.87	92.01	91.73	86.01	91.67	91.35	92.34
		50 43.41	9	10	11	0	0	0	0	14	7	4	3	0	0	0	0	32	10	14	5
	Average		47.15	47.86	48.23	50.36	52.50	52.82	53.42	43.66	49.60	50.21	51.70	50.52	55.54	56.29	57.36	40.50	48.13	48.61	49.27
P			1204.83	1202.08	1181.42	134.75	106.00	79.50	99.42	1632.00	1056.50	1037.00	1111.17	139.17	106.50	69.33	118.58	1744.00	1841.25	1700.17	1527.75

Table 14: Full result of low resource experiment with structural pre-training. For each dataset, upper row is F1 score, and lower row is format errors.