# An Autoregressive Text-to-Graph Framework for Joint Entity and Relation Extraction

**Urchade Zaratiana** [1 2] **Nadi Tomeh** [1] **Pierre Holat** [1 2] **Thierry Charnois** [1]

## Abstract

In this paper, we propose a novel method for joint entity and relation extract from unstructured text by framing it as a conditional sequence generation problem. In contrast to conventional generative information extraction models that generate text as output, our approach generates a linearized graph where nodes represent text spans while the edges/relation of the graph represent relation triples. For that, our method employs a transformer encoder-decoder architecture with pointing mechanism on a dynamic vocabulary of spans and relation types. Particularly, our model can capture the structural characteristics and boundaries of entities and relations through span representation, while simultaneously grounding the generated output in the original text thanks to pointer mechanism. Evaluation on benchmark datasets validates the effectiveness of our approach, demonstrating state-of-the-art results in entity and relation extraction tasks.

## 1. Introduction

Joint entity and relation extraction is a fundamental task in Natural Language Processing (NLP), serving as the basis for various high-level applications such as building Knowledge Graphs. Traditionally, this task was tackled via pipeline models that independently trained and implemented entity recognition and relation extraction, often leading to error propagation (Brin, 1999; Nadeau and Sekine, 2007). Deep learning brought about the development of end-to-end models for this task, enabling shared representations and joint optimization of losses for both tasks (Wadden et al., 2019; Wang and Lu, 2020; Zhong and Chen, 2021; Yan et al., 2021). Despite this advancement, these models essentially remain pipeline-based, with entity and relation predictions executed by separate classification heads, thereby ignoring potential interactions between these tasks.

Recent advancements have seen a shift towards "real" end-to-end solutions, where the prediction of entities and relations is intertwined, accomplished through autoregressive models. These models treat the joint entity-relation task as a process of generating plain text, employing *augmented languages* to encode and decode structured information (Paolini et al., 2021; Lu et al., 2022; Liu et al., 2022; Fei et al., 2022). While these models have achieved remarkable performance, we argue that they also expose room for improvement, especially in terms of grounding.

In this paper, we present an autoregressive transformer encoder-decoder model that generates a linearized graph instead of generating plain text. This approach allows us to capture a more granular interaction between entities and relations and thereby enhances the understanding of their interdependencies. Our model makes use of a pointing mechanism (Vinyals et al., 2017) on a dynamic vocabulary of spans and relations, providing robust grounding in the original text. In fact, without grounding, models can generate outputs that are semantically coherent but contextually detached from the input. Our pointing mechanism mitigates this issue by ensuring that the model's outputs, specifically the entity spans, are directly tied to the input text. Furthermore, by generating spans and relations directly from the text, rather than producing standalone plain text, our model manages to capture the structural characteristics and boundaries of entities and relations more accurately, which can be missed by previous generative information extraction models.

We evaluated our model on three benchmark datasets for joint entity and relation extraction: CoNLL 2004, SciERC, and ACE 05. Our model demonstrated state-of-the-art performance on SciERC and CoNLL 2004 and competitive results on ACE 05. Notably, we achieved a Relation F1 score of 78.5 on the CoNLL 2004 dataset, surpassing the previous best model that scored 76.3. In addition, we also achieved state-of-the-art results on the SciERC dataset. Specifically, our model achieved an Entity F1 score of 69.7 and a Relation F1 score of 38.6 on SciERC, surpassing previous best-performing models by 2.9 and 0.2, respectively.

[1]LIPN, CNRS UMR 7030, France [2]FI Group, France. Correspondence to: Urchade Zaratiana <zaratiana@lipn.fr>.

Accepted to ICML workshop on Structured Probabilistic Inference Generative Modeling

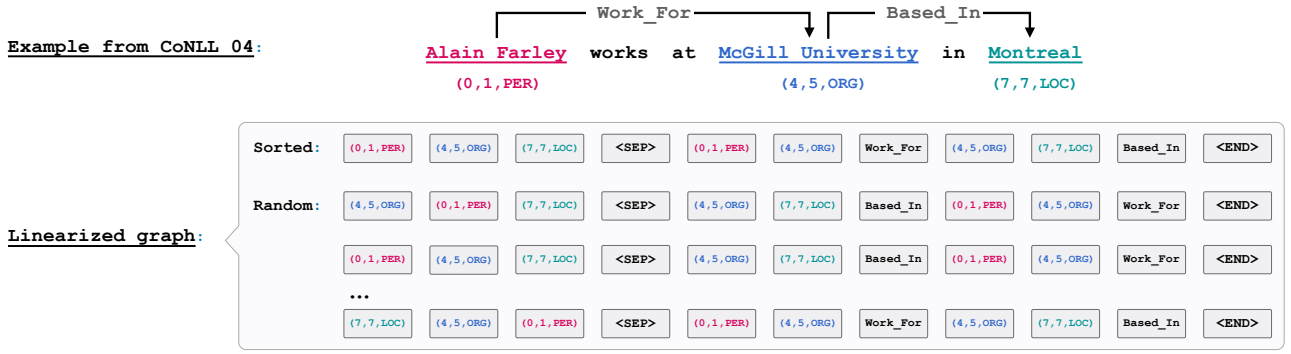


*Figure 1.* **Illustration of the linearization approach for the task of information graph generation**. The input text is mapped into an information extraction graph. The graph consists of entities and relation triples, which are generated sequentially by first producing entity spans (represented by start word, end word, and entity type) followed by relation triples (head entity, tail entity, and relation type). Two different ordering approaches are compared: Sorted and Random. The Sorted approach sorts nodes and edges by the first endpoint of the span and tail entities, respectively. The Random approach randomly orders nodes and edges during training.

## 2. Model

### 2.1. Task definition

In this paper, we address the task of joint entity and relation extraction from text as a graph generation approach. Our proposed model generates nodes and edges as a single sequence, effectively integrating both entity and relation extraction into a unified framework.

Formally, the task can be defined as follows: Given an input text sequence $\mathbf{x} = x_1, x_2, ..., x_L$, where $x_i$ represents the $i$-th token in the sequence, our objective is to generate a linearized graph representation $\mathbf{y} = y_1, y_2, ..., y_M$, where $y_j$ represents a token in the generated sequence. Each token $y_j$ can take one of three forms:

- **Entity Span**: The token $y_j$ represents an entity span, defined by $(s_j, e_j, t_j)$, where $s_j$ and $e_j$ denote the starting and ending positions of the entity span, and $t_j$ denotes the type of the entity.
- **Relation Type**: The token $y_j$ represents a relation type between two entities, such as `Based_In` relation.
- **Special Token**: The token $y_j$ represents special tokens used in the generation process, such as `<SEP>` to separate entities and relations or the `<END>` to stop the generation.

The template employed in our model adheres to a precise sequence generation process, as depicted in Figure 1. Our output sequence consists of the nodes representing the entities, followed by a `<SEP>` token, and ends with the relation triples, each consisting of head node, tail node and edge/relation type. Additionally, we incorporate two distinct approaches for linearization: *sorted* and *random*. The *sorted* linearization organizes entities and relations based on their positions in the original text, while the *random* linearization randomly shuffles entities and relations during each training iteration.

### 2.2. Architecture

**Layers** Our model utilizes a transformer encoder-decoder architecture to process the input text sequence and generate the desired output sequence. The encoder transforms the input sequence $\mathbf{x}$ into token representations $\mathbf{H} \in \mathbb{R}^{L \times D}$, where $D$ is the embedding model dimension. The decoder is trained to predict the subsequent token in the sequence, similar to language modeling. The generation process is conditioned on the previously generated tokens $y_{<j}$ through self-attention and the input token representations $\mathbf{H}$ through cross-attention. The objective of training is to maximize the following conditional probability:

$$\text{maximize} \prod_{j=1}^{M} p(\boldsymbol{y}_j | \boldsymbol{y}_{<j}, \boldsymbol{H})$$

During inference, the model generates the output sequence in an autoregressive manner, one token at a time, using, for instance, *greedy sampling* or *nucleus sampling*.

**Vocabulary construction** Our model uses a pointing mechanism to generate tokens from a dynamic matrix $\mathbf{E}$, which includes spans, special tokens, and relation type embeddings. The latter two are randomly initialized and updated during training. The span embedding, however, is dynamically computed. The embedding of a span

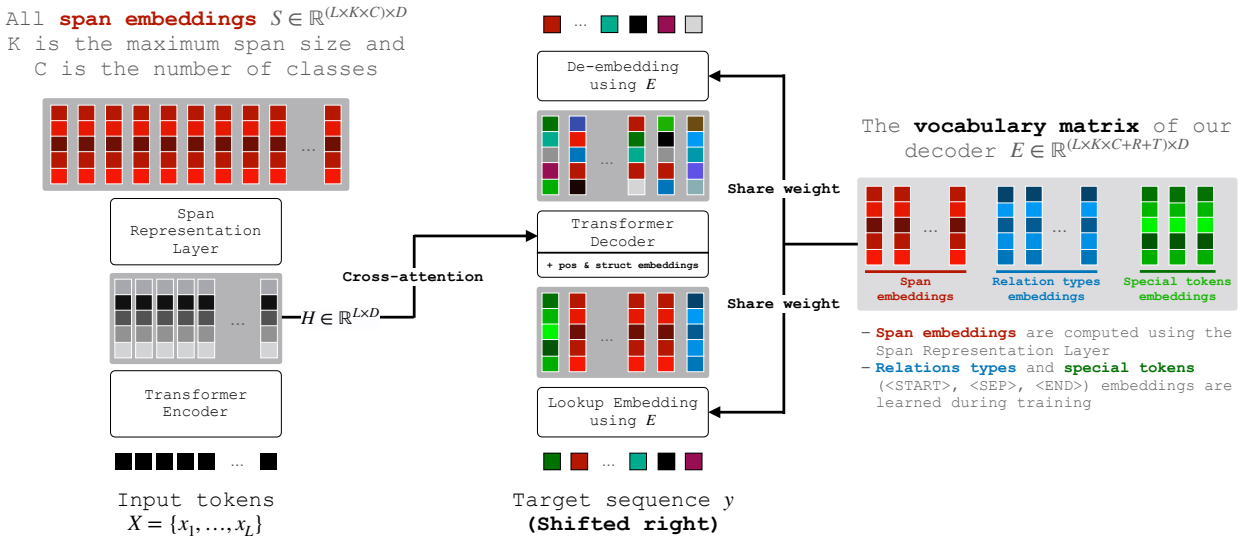


*Figure 2.* **Illustration of the architecture of our model**. (*left*) The Encoder takes in the input sequence $X$ and generates representations of the tokens $\boldsymbol{H}$ and spans $\boldsymbol{S}$. (*middle*) The Decoder then generates the next token conditioned on the previous tokens and the input representation $\boldsymbol{H}$. (*right*) The vocabulary matrix used for decoding consists of the concatenation of span embeddings $\boldsymbol{S}$, learned relation type embeddings, and special token embeddings. This embedding is shared between the input and output embeddings of the Decoder.

$(\text{start}, \text{end}, \text{type})$ is computed as follows:

$$\mathbf{S}[\text{start}, \text{end}, \text{type}] = \mathbf{W}_{\text{type}}^{T}[\mathbf{h}_{\text{start}} \odot \mathbf{h}_{\text{end}}] \tag{1}$$

In this expression, $[\odot]$ represents a concatenation operation, while $\mathbf{h}_{\text{start}}$ and $\mathbf{h}_{\text{end}}$ denote the representations of tokens at the start and end positions, respectively. Finally, $\mathbf{W}_{\text{type}} \in \mathbb{R}^{2D \times D}$ is a weight associated with the entity type. The vocabulary embedding matrix $\mathbf{E}$ is formed by stacking all the span embeddings $\mathbf{S}$, special token embeddings $\mathbf{T}$, and relation type embeddings $\mathbf{R}$.

**Vocabulary Size** The size of the vocabulary matrix $\mathbf{E} \in \mathbb{R}^{V \times D}$ is $V = L \times K \times C + R + T$, where $L$ represents the sequence length, $K$ the maximum span size, $C$ the number of entity types, $R$ the number of relations, and $T$ the number of special tokens (<START>, <END>, and <SEP>). Let's take the CoNLL 2004 dataset as an example to illustrate this. In this dataset, we have the following characteristics: $K = 12$, $C = 4$, $R = 5$, and $T = 3$. Considering a sentence of length 114 (which is the maximum length in the training set), the resulting vocabulary size would be 5480. This size is considerably smaller when compared with the vocabulary size of a typical language model, which usually hovers around 30,000 distinct tokens.

**Decoder modelling** We first embed the previous decoder outputs $y_1, \ldots, y_{i-1}$, using the embedding matrix $\boldsymbol{E}$ (we also add positional and structural embedding, see figure 3):

$$\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{i-1} = \boldsymbol{E}[y_1, \ldots, y_{i-1}] \tag{2}$$

We define $\tilde{\boldsymbol{z}}_i$ as the hidden state at the last position of the decoder output sequence obtained by feeding the previous output embedding and the memory $\boldsymbol{H}$ (for cross-attention) to the decoder:

$$\tilde{\boldsymbol{z}}_i = \texttt{Decoder}(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{i-1}; \boldsymbol{H})[-1] \tag{3}$$

Then, to compute the probability distribution over the vocabulary for generating the next token, $y_i$, we employ the softmax function on the dot product between the dynamic vocabulary embedding matrix $\boldsymbol{E}$ and $\boldsymbol{z}_i$:

$$p(y_i | \boldsymbol{y}_{<i}, \boldsymbol{H}) = \frac{\exp \boldsymbol{E}^T \tilde{\boldsymbol{z}}_i}{\sum_{k=1}^{V} (\exp \boldsymbol{E}^T \tilde{\boldsymbol{z}}_i)_k} \tag{4}$$

The probabilities generated by this formulation allow the model to select the appropriate token from the vocabulary for generating a span, special token, or relation type.

**Constrained decoding** During inference, we employ constrained decoding by enforcing constraints that preserve the well-formedness of the output sequence and ensure its adherence to the original sequence template. To achieve this, we apply masking techniques that restrict the generation of certain tokens during the decoding process.

## 3. Experiments setup

**Datasets** We evaluated our models on three benchmark datasets for joint entity-relation extraction, namely SciERC (Luan et al., 2018), CoNLL 2004 (Carreras and Màrquez, 2004), and ACE 05 (Walker et al., 2006).

| Models | SciERC | | | ACE 05 | | | CoNLL 2004 | | |
|---|---|---|---|---|---|---|---|---|---|
| | ENT | REL | REL+ | ENT | REL | REL+ | ENT | REL | REL+ |
| DYGIE++ (Wadden et al., 2019) | 67.5 | 48.4 | - | 88.6 | 63.4 | - | - | - | - |
| Tab-Seq (Wang and Lu, 2020) | - | - | - | 89.5 | - | 64.3 | 90.1 | 73.8 | 73.6 |
| PURE (Zhong and Chen, 2021) | 66.6 | 48.2 | 35.6 | 88.7 | 66.7 | 63.9 | - | - | - |
| PFN (Yan et al., 2021) | 66.8 | - | 38.4 | 89.0 | - | 66.8 | - | - | - |
| TablERT (Ma et al., 2022) | - | - | - | 87.8 | 65.0 | 61.8 | **90.5** | 73.2 | 72.2 |
| GENERATIVE | | | | | | | | | |
| HySPA (Ren et al., 2021) | - | - | - | 88.9 | 68.2 | - | - | - | - |
| TANL (Paolini et al., 2021) | - | - | - | 89.0 | - | 63.7 | 90.3 | - | 70.0 |
| ASP (Liu et al., 2022) | - | - | - | **91.3** | **72.7** | **70.5** | 90.3 | - | 76.3 |
| UIE (Lu et al., 2022) | - | - | 36.5 | - | - | 66.6 | - | 75.0 | - |
| LasUIE (Fei et al., 2022) | - | - | - | - | - | 66.4 | - | 75.3 | - |
| *Our model* | **69.7** | **51.1** | **38.6** | 90.1 | 68.7 | 66.2 | **90.5** | **78.5** | **78.5** |

*Table 1.* Comparison of our proposed model with state-of-the-art methods on SciERC, ACE 05, and CoNLL 2004 datasets. Results are reported in terms of Entity (ENT) F1, Relation (REL) F1, and Strict Relation (REL+) F1 scores. The best scores are shown in bold, and the second-best scores are underlined.

**Evaluation metrics** Following previous works (Wadden et al., 2019; Fei et al., 2022), for the NER task, we adopt a span-level evaluation requiring precise entity boundaries and type predictions. To evaluate relations, we use two metrics: (1) Boundaries evaluation (**REL**) necessitates the correct prediction of entity boundaries and relations; (2) Strict evaluation (**REL+**) additionally demands accurate entity type prediction. Following previous works, we report the micro-averaged F1 score.

**Hyperparameters** Our model employs an encoder-decoder architecture. We train it for a maximum of 70k steps using AdamW optimizer. We use learning rate warmup for the first 10% of training and then decay to 0. The base learning rates are 3e-5 for the encoder, 7e-5 for the decoder, and 1e-4 for other projecton layers. Our encoders are initialized with pre-trained transformers (Deberta for general domain and SciBERT for scientific datasets). The decoder is randomly initialized with 6 layers, 512 model dimension, and 8 attention heads. We apply sentence augmentation by randomly concatenating sentences from the training set. This allows the model to observe diverse sequence lengths during training and avoids premature generation of the <SEP> token during inference (see Figure 5).

## 4. Results

Our model demonstrates superior results in the Entity and REL+ metrics across the SciERC and CoNLL 2004 datasets. In the Entity recognition task for SciERC, our model achieves the top score of 69.7, surpassing the nearest contender DYGIE++ by 2.2 points. In the strict evaluation (REL+) on the same dataset, our model again leads with a score of 38.6, outstripping the next best model, PFN, by 0.2 points. On CoNLL 2004 dataset, our model excels once more, achieving an Entity recognition score of 90.5. This score is the highest, tied with TablERT, indicating the robustness of our model in identifying and classifying entities accurately. In the more rigorous REL+ metric, our model dominates with a score of 78.5, marking a significant lead over the next best score of 76.3 held by ASP. On the ACE 05 dataset, despite a large parameter model like ASP scoring higher in REL+, our model exhibits comparable performance with an Entity recognition score of 90.1, only 1.2 points behind ASP, and a REL+ score of 66.2.

## 5. Related Works

Joint entity and relation extraction in NLP has traditionally been addressed using pipeline models, leading to error propagation (Brin, 1999; Nadeau and Sekine, 2007). End-to-end models have been developed to jointly optimize entity recognition and relation extraction (Wadden et al., 2019; Wang and Lu, 2020; Zhong and Chen, 2021). However, these models still operate as pipelines, overlooking task interactions.

Recent approaches employ autoregressive models to generate entities and relations as text (Paolini et al., 2021; Lu

et al., 2022; Liu et al., 2022; Fei et al., 2022). We propose an autoregressive transformer encoder-decoder model that generates a linearized graph instead of plain text, capturing fine-grained entity-relation interactions (Vinyals et al., 2017). Our model provides robust grounding in the original text, avoids context detachment, and accurately captures structural characteristics.

## 6. Conclusion

In conclusion, our autoregressive text-to-graph framework for joint entity and relation extraction achieved state-of-the-art results on benchmark datasets. By generating a linearized graph representation instead of plain text, our model accurately captured the structural characteristics and boundaries of entities and relations. The pointing mechanism and dynamic vocabulary provided robust grounding in the original text. The model's superior performance on entity recognition and relation extraction metrics demonstrated its effectiveness in capturing fine-grained interactions.

## Limitations

**Autoregressive Generation** The complexity due to the autoregressive nature of our model can pose problem in term of efficiency, which might make it less suitable for real-time applications.

**Scaling** Our model might struggle to scale to very long sequences, especially with datasets that contain a large number of entity types. This would lead to a substantial increase in vocabulary size.

## References


Sergey Brin. Extracting patterns and relations from the world wide web. In Paolo Atzeni, Alberto Mendelzon, and Giansalvatore Mecca, editors, *The World Wide Web and Databases*, pages 172–183, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg. ISBN 978-3-540-48909-2.

Xavier Carreras and Lluís Màrquez. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-2412.

Hao Fei, Shengqiong Wu, Jingye Li, Bobo Li, Fei Li, Libo Qin, Meishan Zhang, Min Zhang, and Tat-Seng Chua. LasUIE: Unifying information extraction with latent adaptive structure-aware generative language model. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=a8qX5RG36jd.

Tianyu Liu, Yuchen Eleanor Jiang, Nicholas Monath, Ryan Cotterell, and Mrinmaya Sachan. Autoregressive structured prediction with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 993–1005, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.70.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.395. URL https://aclanthology.org/2022.acl-long.395.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. Multi-task identification of entities, relations, and coreferencefor scientific knowledge graph construction. In *Proc. Conf. Empirical Methods Natural Language Process. (EMNLP)*, 2018.

Youmi Ma, Tatsuya Hiraoka, and Naoaki Okazaki. Joint entity and relation extraction based on table labeling using convolutional neural networks. In *Proceedings of the Sixth Workshop on Structured Prediction for NLP*, pages 11–21, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.spnlp-1.2. URL https://aclanthology.org/2022.spnlp-1.2.

David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30:3–26, 2007.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, RISHITA ANUBHAI, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. Structured prediction as translation between augmented natural languages. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=US-TP-xnXI.

Liliang Ren, Chenkai Sun, Heng Ji, and Julia Hockenmaier. HySPA: Hybrid span generation for scalable text-to-graph extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4066–4078, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.356. URL https://aclanthology.org/2021.findings-acl.356.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2017.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span representations. 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 5784–5789, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1585. URL https://aclanthology.org/D19-1585.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. Ace 2005 multilingual training corpus, Feb 2006. URL https://catalog.ldc.upenn.edu/LDC2006T06.

Jue Wang and Wei Lu. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.133. URL https://www.aclweb.org/anthology/2020.emnlp-main.133.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. A partition filter network for joint entity and relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 185–197, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.17. URL https://aclanthology.org/2021.emnlp-main.17.

Zexuan Zhong and Danqi Chen. A frustratingly easy approach for entity and relation extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 50–61, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.5. URL https://aclanthology.org/2021.naacl-main.5.

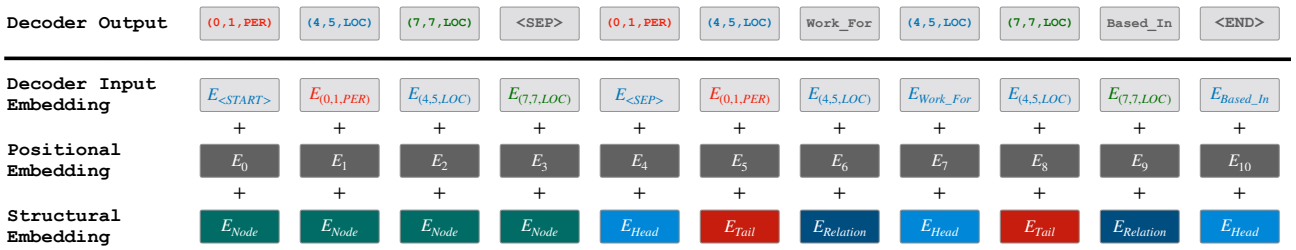


*Figure 3.* **Illustration of our autoregressive generation process for nodes and edges**. The process starts with the special token **<START>** and continues until the **<END>** token is generated. To separate the generation of nodes and edges, a special token **<SEP>** is used. At each position, the decoder takes in the sum of the embedding of the current token, an absolute position embedding, and a structure embedding.

# A. Appendix

## A.1. Ablation studies

**Decoder Input embeddings** As shown in Figure 3, the decoder of our model takes as input the sum of the current token embedding, absolute positional embedding, and structural embedding. The absolute positional embedding is randomly initialized and learned during training. Its goal is to provide positional information to the decoder. The structural embedding tells the model whether it should generate a node ($E_{\text{Node}}$), a head entity ($E_{\text{Head}}$), a tail entity ($E_{\text{Tail}}$), or a relation token ($E_{\text{Relation}}$). Similarly to absolute position encoding, the structural embedding is randomly initialized and updated during training.

| | SciERC | ACE 05 | CoNLL 2004 |
|---|---|---|---|
| Full | **38.6** | **66.2** | **78.5** |
| - Pos | <u>36.4</u> | <u>66.0</u> | 78.3 |
| - Struct | 36.1 | 65.8 | <u>78.4</u> |
| - Both | 35.4 | 65.4 | 78.0 |

*Table 2.* Effect of positional and structural encoding on Relation F1.

The table above (Table 2) shows the ablation of the positional and structural embeddings' effects. When either or both types of encoding are removed, performance drops across all datasets, demonstrating their importance. The effect is most pronounced on the SciERC dataset.

**Sequence Ordering** Our experimental analysis reveals that the sequence ordering of entity and relation tokens significantly impacts the performance of our model in generating output sequences for entity and relation extraction. Preserving the sorted order of these tokens generally leads to improved performance on the SciERC and ACE 05 datasets. However, the influence of sequence ordering is less pronounced on the CoNLL 2004 dataset.

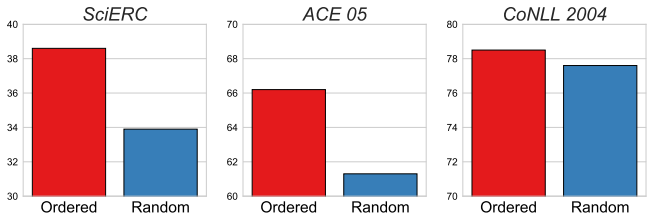


*Figure 4.* Impact of sequence ordering (Ordered and Random) on F1 Relation performance

*Figure 5.* Sentence Augmentation

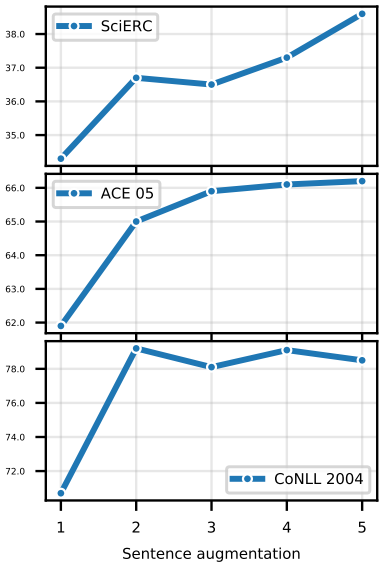


*Figure 6.* Nucleus Sampling Top-p

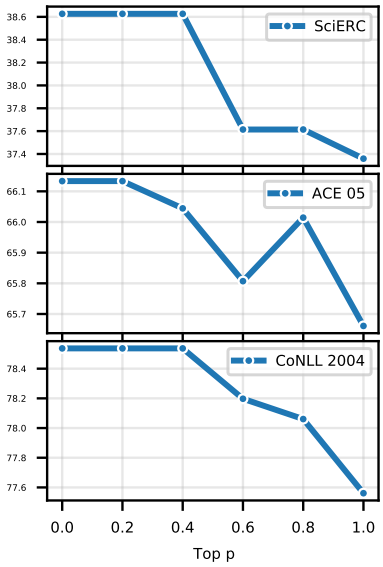


*Figure 7.* Number of decoder layers