Information Extraction with Differentiable Beam Search on Graph RNNs

Niama El Khbir, Nadi Tomeh, Thierry Charnois
LIPN, CNRS UMR 7030, France
{elkhbir,tomeh,charnois}@lipn.fr

Abstract

Information extraction (IE) from text documents is an important NLP task that includes entity, relation, and event extraction. These tasks are often addressed jointly as a graph generation problem, where entities and event triggers represent nodes and where relations and event arguments represent edges. Most existing systems use local classifiers for nodes and edges, trained using cross-entropy loss, and employ inference strategies such as beam search to approximate the optimal graph structure. These approaches typically suffer from exposure bias due to the discrepancy between training and decoding. In this paper, we tackle this problem by casting graph generation as auto-regressive sequence labeling and making its training aware of the decoding procedure by using a differentiable version of beam search. We evaluate the effectiveness of our approach through extensive experiments conducted on the ACE05 and ConLL04 datasets across diverse languages. Our experimental findings affirm that our model outperforms its non-decoding-aware version for all datasets employed. Furthermore, we conduct ablation studies that emphasize the effectiveness of aligning training and inference. Additionally, we introduce a novel quantification of exposure bias within this context, providing valuable insights into the functioning of our model.

Keywords: Information Extraction, Differentiable Beam Search, Auto-regressive Sequence Labeling

1. Introduction

Information extraction (IE) is a crucial task in natural language processing that involves identifying and labeling salient *entities* and semantic *relations* between them, *triggers* of events and their arguments which are entities that play specific *roles* in the event. The output is often formalized as a *labeled graph* where entities and triggers are represented by nodes, relations by edges joining two entity nodes, and roles by edges joining a trigger node and an entity node. See Figure (1) for an example graph for an input text and (§2) for formal definitions.

An important challenge is to adequately model the dependencies between labels. Many approaches have been studied in the literature including modeling inter-instance and inter-label dependencies using a globally-normalized CRF-based scoring function (Zheng et al., 2017). Another line of work uses auto-regressive frameworks that take previous decisions into account to construct representations for next predictions. This includes the work of (Luan et al., 2019) and (Wadden et al., 2019) which uses graph convolution layers to iteratively refine node representations but still use independent classifiers for labeling. Other auto-regressive frameworks rely on sequence labeling models with RNNs (Zheng et al., 2017) or on Seq2Seq models with Transformers (Paolini et al., 2021; Lu et al., 2022; Fei et al., 2022; Liu et al., 2022; Zaratiana et al., 2024) that use specifically-designed vocabularies to encode the labeled graph.

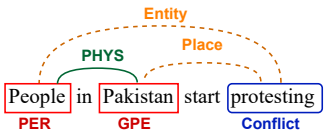

Figure 1: Example of an IE graph. Entity nodes are framed in red, trigger nodes in blue, relation edges in green and argument edges in orange.

Training such auto-regressive sequence models typically consists of maximizing the locally normalized likelihood of each token in the reference (gold) sequence given previous reference tokens. For inference, the unknown previous tokens are replaced by model predictions which create a discrepancy, which results in exposure bias and error propagation. Existing solutions, such as schedule sampling (Bengio et al., 2015), incorporate previous decoding decisions stochastically during training. However, the training objective becomes discontinuous because it relies on greedy decisions at each time step, hence hindering gradient-based learning. Furthermore, when beam search is used instead of greedy decoding the objective does not directly *reason* about the behavior of the decoder at inference time. As a result beam decoding can sometimes yield reduced test performance when compared with greedy decoding (Cho et al., 2014; Koehn and Knowles, 2017). Previous work proposed a training objective that takes search into consideration by proposing continuous approximations of both greedy and beam search decoding (Goyal et al., 2017, 2018). This makes the de-

LREC-COLING 2024, pages 9084–9096
20-25 May, 2024.

coding stage differentiable, hence allowing it to be used in gradient-based learning. This approach makes the model aware of the decoding process during training, resulting in better performance which has been shown to help sequential tasks such as named-entity recognition and segmentation, but it has not been applied to more general graph generation.

In this work, we reformulate labeled IE graph generation as a sequence labeling problem. Similar to previous work (Lin et al., 2020), we first identify entities and triggers using a linear-chain CRF (Lafferty et al., 2001) with a BIO tagging scheme, then autoregressively decode the output graph by sequentially labeling identified nodes and possible edges between them. Unlike Lin et al. (2020) which uses a combination of local classifiers with manually designed feature-based representation of the graph, we apply RNNs on the *linearized* graph (You et al., 2018a). We use beam search for labeling and show that the discrepancy between training and decoding is harmful. To solve this issue we propose a continuous relaxation of beam decoding similar to Goyal et al. (2018). We conduct experiments on the ACE05 and CoNLL04 datasets, spanning multiple languages. Through these experiments, we demonstrate the superiority of our model over its non-decoding-aware counterpart (§5). In addition, we perform ablations studies confirming that the best performance is obtained when training and inference are aligned (§6.1 and §6.2). Finally, we propose a quantification of exposure bias (§6.3), offering deeper insights into our model.

2. Task Definition

Information extraction involves identifying and labeling entities, relations, triggers, and their arguments in text data, mapping it to a labeled graph $G = (V, E)$. V is the set of nodes corresponding to entities and triggers, and E is the set of edges corresponding to relations between pairs of entities or between a trigger and one of its arguments. Each graph element (a node or an edge) is assigned a label from a set of possible types. Figure 1 represents an example of an IE graph.

3. Model

The model we propose is composed of two systems trained in a multitask fashion. The first system focuses on identifying nodes of the graph using a CRF for sequence labeling (§3.1). The second system tackles the generation of the labeled graph using an auto-regressive network. It takes the identified nodes from the previous step, produces a linearized graph over them (§3.2), and labels the resulting sequence using RNNs for representation and beam search for decoding (§3.3). In §3.4 we describe the relaxation of the beam search which we use for our search-aware training procedure.

3.1. Nodes Identification

Text Encoding The input sequence is passed through a pretrained language model (PLM), such as BERT (Devlin et al., 2019a), to generate a vector representation for each word in the sequence. If a word is split into multiple word pieces, we consider its representation to be the average of all its word piece vectors.

Identification as Sequence Labeling The sequence of embeddings is passed through a feed-forward layer and then fed to a CRF (Lafferty et al., 2001) layer. The CRF labels the sequence using the BIO scheme to identify spans of tokens corresponding to entities or triggers. We use two separate CRFs to allow overlapping between entities and triggers. Referring to the example in Figure 1, the entity CRF yields the sequences <B, O, B, O, O>, the trigger CRF yields <O, O, O, O, B>.

Training and Inference During training, we use the negative log-likelihood L_{id} of the reference BIO tag sequence as the loss function. This loss function is part of the overall joint-training loss of our model. For inference, we employ the Viterbi algorithm to search for the most likely tag sequence.

3.2. Graph Linearization

The graph consists of nodes denoted by $V = \{e_1, \ldots, e_n, t_1, \ldots, t_m\}$, representing the previously identified entities and triggers. Entities are arranged in the order of their appearance in the sentence as $e_1, \ldots, e_n$, while triggers follow a similar ordering as $t_1, \ldots, t_m$. To predict the types of entities, triggers, relations, and arguments, we consider all possible pairwise relations and arguments $E = \{(e_i, e_j) \in V^2\}_{1 \le i < j \le n} \cup \{(t_i, e_j) \in V^2\}_{\substack{1 \le i \le m \\ 1 \le j \le n}}$. These pairs are treated as an ordered sequence using lexicographic order.

We construct the linearized graph sequence using the entity, relation, trigger, and argument sequences according to the following procedure: we iterate over the entity sequence, and at each step, we add the current entity and all relations between it and the previously added entities. This ensures that each relation appears after its two endpoints. Subsequently, we iterate over the trigger sequence, adding at each step the visited trigger and then all possible arguments. The resulting sequence is of length $T = n + \frac{n(n-1)}{2} + m + nm$, where n, m, $\frac{n(n-1)}{2}$, and nm respectively represent the number

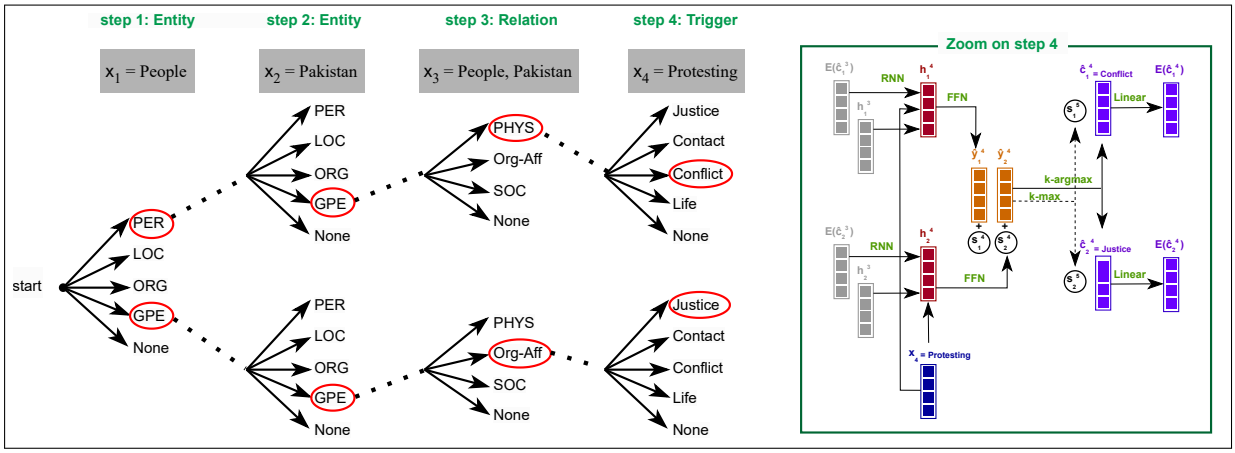

Figure 2: SLBS example for Figure 1, $K = 2$, $|\mathcal{V}_{entity}| = |\mathcal{V}_{trigger}| = 5$, and $|\mathcal{V}_{relation}| = 4$. Hidden state h_1^3 encodes the following graph path tags: *PER, GPE, PHYS* and h_2^3 encodes: *GPE, GPE, Org-Aff*.

of entities, triggers, relations, and arguments. The sequence follows the ordering: e_1, e_2, (e_1, e_2), e_3, $\ldots$, (e_{n-1}, e_n), t_1, (t_1, e_1), (t_1, e_2), $\ldots$, (t_m, e_n).

As an example, the linearization of the graph in Figure 1 results in: *"People"*,*"Pakistan"*, (*"People", "Pakistan"*), *"Protesting"*, (*"Protesting", "People"*), (*"Protesting", "Pakistan"*).

During training, this sequence is constructed using gold entities and gold triggers from the input sentence.

3.3. Graph RNN with Beam Search

Encoding of Nodes and Edges A node's representation is the average of its token representations. An edge representation is a concatenation of its two node representations. We denote the encoded sequence as $x \in \mathbb{R}^{d_x \times T}$, and for ease of readability, we denote x^i its i-th element in all the following.

Labeling Given the linearized graph $\mathbf{x}$, we aim to generate a label sequence $\hat{\mathbf{c}}$ of the same length, where each element x^i is assigned a label from the corresponding task vocabulary $\mathcal{V}_{task}$, where $task$ is one of the four IE tasks (entity, relation, trigger, and role). For each $\mathcal{V}_{task}$, we add a dedicated `None` label for graph elements that have to be removed from the graph.

Sequence Labeling with Beam Search (SLBS), with a beam size K, is a heuristic that approximates the most likely label sequence by keeping track of and updating K candidate sequences at each step. At each step $t = \{1, \ldots, T\}$, we keep track of K couples $\{(h_i^t \in \mathbb{R}^{d_h}, s_i^t \in \mathbb{R})\}_{1 \leq i \leq K}$. The vector h_i^t can be understood as an embedding of the i-th beam element of the usual beam search algorithm, and is updated using a Recurrent Neural Network (RNN):

$$h_i^{t+1} = \text{RNN}(x^{t+1}, \mathcal{E}(\hat{c}_i^t), h_i^t) \quad (1)$$

where $x^{t+1} \in \mathbb{R}^{d_x}$ is the current instance embedding, $\mathcal{E}(\hat{c}_i^t)$ is the embedding of $\hat{c}_i^t$ implemented as a linear projection layer, with $\hat{c}_i^t \in \{1, \ldots, |\mathcal{V}_{task}|\}$ being the index of the previously selected tag, i.e. the one that best extends the i-th element of the beam. The term "best" is defined in this context using the extension scores $\tilde{s}_{i,j}^t \in \mathbb{R}$ such that, for every beam index $1 \leq k \leq K$:

$$s_k^{t+1} = \underset{\substack{1 \leq i \leq K \\ 1 \leq j \leq |\mathcal{V}_{task}|}}{\texttt{top-k-max}}(\tilde{s}_{i,j}^t) \quad (2)$$

$$b_k^t, \hat{c}_k^t = \underset{\substack{1 \leq i \leq K \\ 1 \leq j \leq |\mathcal{V}_{task}|}}{\texttt{top-k-argmax}}(\tilde{s}_{i,j}^t) \quad (3)$$

With

$$\tilde{s}_{i,j}^t = s_i^t + \hat{y}_{i,j}^t \quad (4)$$

The local scores $\hat{y}_{i,j}^t \in \mathbb{R}$ represent classification logits produced by feed forward networks FFN_{task} when fed the hidden states h_i^t:

$$\hat{y}_{i,\cdot}^t = \text{FFN}_{task}(h_i^t) \in \mathbb{R}^{|\mathcal{V}_{task}|} \quad (5)$$

The local score $\hat{y}_{i,j}^t$ can be seen as the negative log-likelihood of the beam element i having j as a tag at time step t.

In equation 3, $b_k^t \in \{1, \ldots, K\}$ serve as back-pointers because they point to the beam element whose extension produced the current state of the beam element k.

In practice, updates are made in the following order: 5, 4, 3, 2 / 1. Figure 2 illustrates an example of the first 4 steps of the SLBS procedure.

Algorithm 1 Soft SLBS training for IE

Input: $x = x^1, \ldots, x^T$, the linearized graph

1 **for** *t=1 to T* **do**
2 **for** *i=1 to K* **do**
3 $h_i^t \leftarrow \text{RNN}(x^{t+1}, \mathcal{E}(\hat{c}_{\cdot,i}^t), h_i^t)$
4 **for** *j=1 to* $|\mathcal{V}_{task}|$ **do**
5 $\hat{y}_{i,j}^t \leftarrow \text{FFN}_{task}(h_i^t)_j$
6 $\tilde{s}_{i,j}^t \leftarrow s_i^t + \hat{y}_{i,j}^t$
7 **for** *j=1 to* $|\mathcal{V}_{task}|$, *k=1 to K* **do**
8 $w_{i,j}^k \leftarrow (\tilde{s}_{i,j}^t - \underset{\substack{1\le i\le K \\ 1\le j\le|\mathcal{V}_{task}|}}{\texttt{top-k-max}}(\tilde{s}_{i,j}^t))^2$
9 $p_{i,j}^k \leftarrow \sigma(\frac{-w_{\cdot,\cdot}^k}{\alpha})_{i,j}$
10 **for** *k=1 to K* **do**
11 $s_k^{t+1} \leftarrow \sum_{i,j} p_{i,j}^k \tilde{s}_{i,j}^t$
12 **for** *j=1 to* $|\mathcal{V}_{task}|$, *k=1 to K* **do**
13 $\hat{c}_{j,k}^t \leftarrow \frac{\sum_i p_{i,j}^k}{i}$
14 $loss+ = \sum_k d_k((-\log(\sum_i e^{-w_{i,j_t^*}^k + a}) + a) + (\log(\sum_{i,j_t^*} e^{-w_{i,j_t^*}^k + b}) - b)$
 with $a = \min_i(w_{i,j_t^*}^k)$ and $b = \min_{i,j}(w_{i,j_t^*}^k)$

Training During training, K $= 1$. Hence, the model is greedily trained to minimize the total cross-entropy L_g loss at each time step between the predicted tags and the gold ones:

$$L_g = -\sum_{t=1}^{T}\sum_{j=1}^{|\mathcal{V}_{task}|} y_j^t \log(\sigma(\hat{y}_{i,j}^t)) \quad (6)$$

Where $y_\cdot^t \in \mathbb{R}^{|\mathcal{V}_{task}|}$ is the gold tag in its one-hot form, and σ is the softmax function.

Total Loss The model is jointly trained to minimize the nodes identification loss and the labeled graph generation loss: $L = L_{id} + L_g$.

3.4. Continuous Relaxation of Beam Search

The SLBS procedure is used as a decoding strategy with models that are trained greedily using cross-entropy. Hence, the distribution of hidden states reached during inference does not match that of the hidden states reached during training. In order to incorporate awareness of the decoding strategy into the training stage, we train our model using a relaxed SLBS procedure, by replacing the discontinuous `top-k-argmax` operation with the relaxed version used by Goyal et al. (2018); Maddison et al. (2017); Jang et al. (2017); Goyal et al. (2017) in the context of Seq2Seq models.

The following describes how we relax the SLBS procedure for IE, making it fully continuous and almost everywhere differentiable.

Continuous `top-k-argmax` The key ingredient is to replace the only discontinuous operation of the SLBS procedure, namely the `top-k-argmax` operation applied to extension scores, with a continuous approximation, taking advantage of the following asymptotic property: for any real-valued function f defined over the vocabulary $\mathcal{V}_{task}$, the expression $\sigma(-\frac{(f(\cdot)-m_k)^2}{\alpha})_j = \frac{e^{\frac{-(f(j)-m_k)^2}{\alpha}}}{\sum_{l=1}^{|\mathcal{V}_{task}|} e^{\frac{-(f(l)-m_k)^2}{\alpha}}}$ tends to $\delta_j(\underset{1\le l\le|\mathcal{V}_{task}|}{\texttt{top-k-argmax}}(f(l)))$ as the temperature parameter α tends to zero, with δ_j being the Dirac distribution centered on the tag j, which can also be seen as the one-hot operation, and:

$$m_k = \underset{1\le l\le|\mathcal{V}_{task}|}{\texttt{top-k-max}}(f(l)) \quad (7)$$

Training with soft SLBS In the SLBS procedure, the `top-k-argmax` operation is used to make tag choices $\hat{c}_k^t$ based on the extension scores $\tilde{s}_{i,j}^t$. In the relaxed setup, a tag choice is no longer a binary decision. Therefore, using the previous asymptotic approximation, we define $p_{i,j}^k$ as the set of probability distributions over tags j (cf. lines 8 and 9 of Algorithm 1) that can be interpreted as the probability of beam element k being updated using the hidden state coming from beam element i and extended by tag j.

Such a set of probability distributions can be first used to compute a relaxed version of s_k^{t+1}, as the expected extension score over all origin beam elements i and extension tags j (cf. line 11 of Algorithm 1), and then to compute a relaxed version of the one-hot representation of the previously added tag $\hat{c}_k^t$, denoted $\hat{c}_{j,k}^t$, as the probability of j being the last tag added to the beam element k (cf. line 13 of Algorithm 1).

Loss Computation Importantly, this set of probability distributions can be used to compute the negative log-likelihood of each tag in the gold sequence, which is a problem-adapted local loss:

$$l^t = -\log P(j_*^t) = -\log(\sum_{k=1}^{K} d_k(\sum_{i=1}^{K} p_{i,j_*^t}^k)) \quad (8)$$

where j_*^t denotes the index of the gold tag at time step t, $\sum_{i=1}^{K} p_{i,j_t^*}^k$ represents the (marginalized) probability of j_*^t being the predicted tag given a beam k, that also can be interpreted as a posterior over the set of beams $1, \ldots, K$, and d_k being a prior over the set of beam elements. Overall, we

associate the labeled graph generation with the following global loss:

$$L_{cl} = \sum_{t=1}^{T} l^t \quad (9)$$

Unfortunately, empirical observations show numerical instability in the computation of l^t. To address this issue, one possible approach is to tightly bound it with a term that can be stabilized using techniques such as the log-sum-exp trick. Note that the earlier trick cannot be directly applied to l^t due to the sum $\sum_{k=1}^{K}$ being inside the $\log$. Additionally, we must consider the trade-off between the stable upper bound and l^t (referred to as the stabilization margin), as a larger gap between them implies a greater misalignment between the training and inference procedures. Thus, instead of minimizing l^t, we minimize the quantity presented in line 14 of Algorithm 1.

Total Loss The model is jointly trained to minimize the nodes identification loss and the labeled graph generation loss: $L = L_{id} + L_{cl}$.

4. Experimental Setup

4.1. Datasets

We evaluate our model on 2 datasets and 3 different languages: ACE05 (Walker and Consortium, 2005) for English, Arabic, and Chinese, and CoNLL04 (Roth and Yih, 2004). For English ACE05, we consider two versions from the literature: ACE05-R, which involves entity and relation extraction, and ACE05-E+, which includes entity, relation, and event extraction. We follow the data splits and preprocessing of Luan et al. (2019) and Lin et al. (2020) for ACE05-R and ACE05-E+. For Chinese data, we use the same preprocessing and splits of Lin et al. (2020) and refer to it by ACE05-CN. For Arabic data, we use the same preprocessing and splits of El Khbir et al. (2022) and refer to it by ACE05-AR. Thus, CoNLL04 involves 4 entity types and 5 relation types, and ACE05 involves 7 entity types, 6 relation types, 33 event types, and 22 argument types. Table 1 provides statistics of the datasets.

4.2. Evaluation Metrics

We evaluate our model using micro F1 measure. An entity prediction or an event trigger prediction is considered correct if its type and boundaries match those of the gold one. For relations and event arguments, we adopt the boundaries evaluation (Taillé et al., 2020), a nonstrict and undirected evaluation, where a relation or an argument is considered correct if its type and boundaries align with the gold one. Additionally, we report the average F-scores across all tasks to evaluate the model globally. We average scores from three runs and report numbers for the model with the highest average F1 on the dev set.

Dataset	Split	SENT	ENT	REL	EVT	ARG
ACE05-R	Train	10,051	26,473	4,788	-	-
	Dev	2,424	6,338	1,131	-	-
	Test	2,050	5,476	1,151	-	-
CoNLL04	Train	922	3,377	1,283	-	-
	Dev	231	893	343	-	-
	Test	288	422	422	-	-
ACE05-E	Train	19,240	47,554	7,159	4,419	6,607
	Dev	901	3,423	728	468	759
	Test	676	3,673	802	424	689
ACE05-CN	Train	6,841	29,657	7,934	2,926	5,463
	Dev	526	2,250	596	217	403
	Test	547	2,388	672	190	332
ACE05-AR	Train	2,936	26,031	3,712	1,830	3,176
	Dev	382	3,256	498	234	401
	Test	371	2,925	392	204	334

Table 1: Number of sentences (i.e., **SENT**), entities (i.e., **ENT**), relations (i.e., **REL**), event triggers (i.e., **EVT**) and event arguments (i.e., **ARG**).

4.3. Settings and Hyperparameters

For the PLMs, we use *bert-large-cased* (Devlin et al., 2019a) for ConLL04, ACE05-R, and ACE05-E+, *bert-large-arabertv2* (Antoun et al., 2020) for ACE05-AR, and *bert-large-chinese* for ACE05-CN. We fine-tune the hyperparameters on ACE05-E+ and use the same settings for other ACE05 data. We search for *K* values in {4, 10, 16, 20, 22} and we retain the model with $K = 10$. We search for α values in {0.1, 0.5, 1, 2, 5, 10} and retain α=1. We use for d_k the uniform prior. We ran our experiments on a GPU Nvidia GEForce RTX 2080 with 8 GB of RAM. We estimate the needed computational budget for each training epoch to be 3, 10, 20, 6, and 5 GPU minutes respectively for ConLL04, ACE05-R, ACE05-E+, ACE05-AR, and ACE05-CN. The hyperparameters used include Adam optimizer, BERT learning rate 1e-5, BERT weight decay 1e-5, BERT dropout 0.5, gradient clipping 5.0, learning rate 1e-4, weight decay 1e-4, dropout 0.4, and hidden sizes of 256 for the RNN, 150 for FFN_{node}, and 600 for FFN_{edge}.

5. Results and Analysis

Main Results The main results of our experiments on ConLL04 and ACE05 data, along with some literature results, are presented in Tables 2 and 3. We begin by establishing a baseline with the Sequence Labeling Beam Search model (SLBS), trained in a greedy way and decoded using beam search (§3.3). We then present the results of the

Model	CoNLL04			ACE05-R			ACE05-E+				
	ENT	REL	AVG	ENT	REL	AVG	ENT	REL	EVT	ARG	AVG
Wang and Lu (2020)$^{\times}$	**90.1**	**73.8**	**81.9**	89.5	67.6	78.5	-	-	-	-	-
Wadden et al. (2019)* +	-	-	-	88.4	63.2	75.8	-	-	-	-	-
Zhong and Chen (2021)*	-	-	-	88.7	66.7	77.7	-	-	-	-	-
Ye et al. (2022)*	-	-	-	**89.8**	69.0	**79.4**	-	-	-	-	-
Zhang and Ji (2021)†	-	-		88.7	67.2	77.9	91.0	62.8	72.7	57.7	71.0
Nguyen et al. (2022b)†	-	-	-	-	-	-	**91.7**	64.9	74.6	**61.2**	**73.1**
Lin et al. (2020)$^{\diamond}$	-	-	-	88.8	67.5	78.1	89.6	58.6	72.8	54.8	69.0
Nguyen et al. (2021)$^{\diamond}$	-	-	-	88.9	68.9	78.9	91.1	63.6	73.3	57.5	71.4
Nguyen et al. (2022a)$^{\diamond}$	-	-	-	88.9	**69.5**	79.2	91.0	**65.4**	74.8	59.9	72.7
SLBS $^{\diamond}$	90.0	68.6	79.4	88.9	68.2	78.6	91.4	63.8	73.3	55.6	71.0
SSLBS $^{\diamond}$	90.1	71.4	80.8	88.5	69.2	78.9	91.2	64.0	**75.0**	56.9	71.8

Table 2: Performance on English. Models grouped in the same group of rows use the same encoder for word representations; $\times$: *albert-xxlarge*, $*$: *bert-base*, †: *roberta-large*, $\diamond$: *bert-large*. Models marked with a + sign use extra training data.

ACE05-CN					
Model	**ENT**	**REL**	**EVT**	**ARG**	**AVG**
Lin et al. (2020)$^{\diamond}$	88.5	62.4	65.6	52.0	67.1
Nguyen et al. (2021)$^{\diamond}$	88.7	65.1	66.5	54.9	68.8
Nguyen et al. (2022b)*	**89.2**	**68.3**	**74.3**	**60.0**	**72.9**
SLBS†	88.6	64.8	65.9	49.6	67.3
SSLBS†	**89.2**	67.1	68.3	52.4	69.3
ACE05-AR					
Model	**ENT**	**REL**	**EVT**	**ARG**	**AVG**
El Khbir et al. (2022)$^{\times}$	85.1	62.9	63.6	51.8	66.0
SLBS$^{\times}$	**85.3**	**63.1**	62.0	51.6	65.5
SSLBS$^{\times}$	84.6	**63.1**	**63.9**	**55.0**	**66.6**

Table 3: Performance on Chinese and Arabic. $\diamond$:*bert-multilingual-cased*, $*$:*xlm-roberta-large*, †:*bert-large-chinese*, $\times$:*bert-large-arabertv2*

Soft SLBS (SSLBS) model trained with relaxed beam search and decoded using beam search (§3.4).

The results show that the SSLBS model improves the baseline average F-score across all used datasets. Specifically, the SSLBS model demonstrates improvements of 1.4, 0.3, 0.8, 2.0, and 1.1 F-score points on ConLL04, ACE05-R, ACE05-E+, ACE05-CN, and ACE05-AR, respectively. This suggests that the decoding-aware training strategy is indeed more effective than greedy training.

Comparison to Other Works For English, we compare our model to Lin et al. (2020), Nguyen et al. (2021), and Nguyen et al. (2022a) since we use the same PLM as an encoder. Among these works, SSLBS has the second-best relation and average F-scores on ACE05-R, the best trigger F-score, and the second-best entity, relation, and average F-score on ACE05-E+. In addition, we consider other joint IE models such as Wadden et al. (2019); Zhang and Ji (2021); Nguyen et al. (2022b), as well as models that focus solely on joint ERE (Wang and Lu, 2020; Zhong and Chen, 2021; Ye et al., 2022). While these models employ various techniques such as span graph propagation (Wadden et al., 2019), manually-designed global features (Lin et al., 2020; Zhang and Ji, 2021), global type dependency regularization (Nguyen et al., 2021), and dependency-induced graphs with simulated annealing (Nguyen et al., 2022a), the SSLBS model implicitly learns graph representations through the hidden states of the network.

For Arabic and Chinese, SSLBS exhibits comparable performance to other existing approaches, with the trigger and argument tasks showcasing substantial performance gains.

Overall, while SSLBS does not surpass all SOTA models, it still achieves competitive scores. To ensure fairness in comparisons, evaluating with the same PLM is preferable (Taillé et al., 2020). However, the focus of our work is on integrating the decoding procedure into training, rather than exploring different PLM parameters. We make our code publicly available for further investigations.

6. Ablation Studies

6.1. Effect of Forward/ Prediction Beam Sizes

To ensure alignment between training and inference objectives, we investigate the impact of different beam sizes on our model's performance. We denote here *fbs*, the forward beam size used during training, and *pbs* the prediction beam size used during inference. Figure 3 shows the obtained average F-scores, with a fixed temperature α=1, for ACE05-E+ dataset.

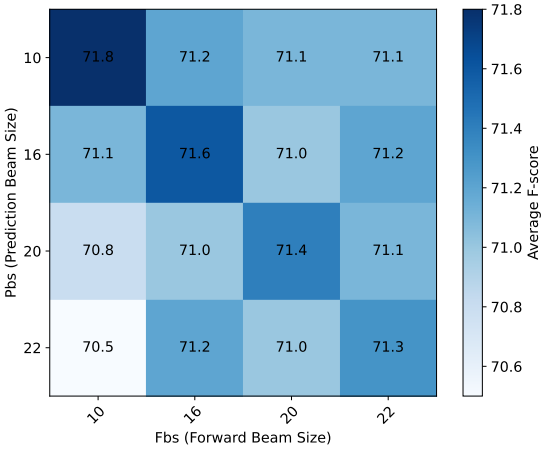

Figure 3: Effect of *fbs* and *pbs* on performance - ACE05-E+ data.

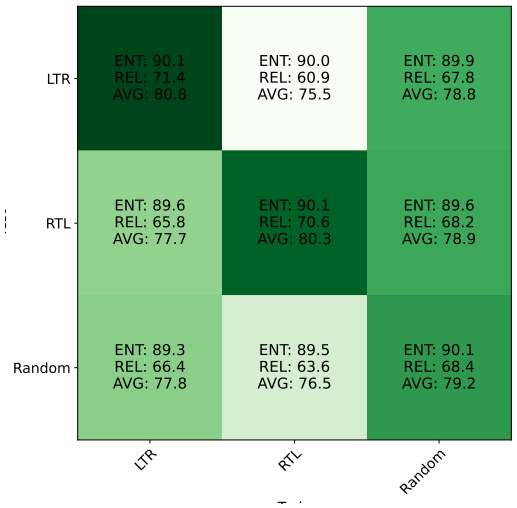

Figure 4: Effect of sequence ordering on performance - CoNLL04 data.

We notice that the diagonal of the matrix, corresponding to *fbs*=*pbs*, is prevailing. This indicates that the model achieves its best results when the training closely aligns with the inference process.

In addition, we notice that the scores of the over-diagonal, corresponding to *fbs* > *pbs*, consistently outperform those of the under-diagonal, corresponding to *fbs* < *pbs*. This suggests that a model trained with a larger beam size has a broader exposure to potential options during training, enabling it to better handle search errors that occur when decoding with a smaller beam size. Conversely, the lowest score is obtained for the $\{fbs = 10, pbs = 22\}$ combination, which highlights a performance decline when the beam size used during decoding is larger than that during training. These insights emphasize the importance of aligning beam sizes to enhance model performance and generalization.

6.2. Effect of Sequence Ordering

We perform experiments to explore the impact of varying sequence orders during both the training and testing phases. For all previous experiments, we have adhered to the sequence order outlined in §3.2, denoted here as the left-to-right (LTR) order. However, to comprehensively assess our model's performance, we introduce two alternative sequence orders: the right-to-left (RTL) order and a random (Random) order. In the RTL order, we maintain fixed node positions while rearranging the edges in a right-to-left fashion. Conversely, the Random order involves a random reordering of edges while keeping node positions constant. We conduct these experiments on ConLL04, and the results are depicted in Figure 4.

We observe a dominant trend along the diagonal in Figure 4, which indicates that the model consistently excels when tested on the same order it was trained on, thus when training and inference are aligned. Notably, training and testing with the LTR ordering consistently yield the best performance, possibly because the LTR order aligns well with the natural sequential dependencies of the data.

Additionally, training the model with the Random order and testing it with different orders (last column) demonstrates superior adaptability and robustness compared to training with either LTR or RTL. The model's ability to adapt to novel sequence arrangements stands out in this scenario.

6.3. Exposure Bias Quantification

We assess exposure bias in two settings: SLBS and SSLBS (with various temperature α values). We also explore the use of Teacher Forcing (TF) and model predictions (no TF) in both settings. We conduct these experiments on ConLL04, training the model for 150 epochs and reporting results of the last epoch in Table 4.

Exposure Bias Definition Exposure bias (EB) refers to the gap between training and testing conditions for a model. We quantify EB by computing the Kullback-Leibler divergence between the distributions of training hidden states $P_{h_{\text{train}}}$ and decoding hidden states $P_{h_{\text{test}}}$. We practically compute this divergence using an N-samples Monte-Carlo scheme:

$$D_{\text{KL}}(P_{h_{\text{train}}} \,||\, P_{h_{\text{test}}}) \underset{h_i \sim P_{h_{\text{train}}}}{\approx} \frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{P_{h_{\text{train}}}(h_i)}{P_{h_{\text{test}}}(h_i)} \right) \quad (10)$$

MODEL	TF					no TF				
	SLBS	SSLBS $\alpha = 0.01$	SSLBS $\alpha = 0.1$	SSLBS $\alpha = 1$	SSLBS $\alpha = 10$	SLBS	SSLBS $\alpha = 0.01$	SSLBS $\alpha = 0.1$	SSLBS $\alpha = 1$	SSLBS $\alpha = 10$
EB	258.7	146.2	142.4	25.7	82.3	163.3	544.6	39.6	3.7	112.0
FVC	192	8	15	21	30	35	15	13	15	22
ENT	89.4	90.2	90.1	90.3	89.9	90.1	90.0	90.4	90.3	89.5
REL	69.5	66.7	68.3	71.3	70.3	69.2	67.3	68.3	71.3	67.6
AVG	79.4	78.4	79.2	80.8	80.1	79.6	78.6	79.3	80.8	78.6

Table 4: Exposure Bias Quantification. TF: Teacher Forcing, EB: Exposure Bias Values. FVC: Features Vectorial Complexity.

Besides, we approximate these hidden state distributions $P_{h_{\text{train}}}$ and $P_{h_{\text{test}}}$ as Gaussian Mixtures (Reynolds, 2009), using 5 components.

Dimensionality Reduction and Feature Vectorial Complexity Empirical observations suggest that trained models often make little to no use of certain hidden state dimensions. To streamline calculations and reduce noise in hidden states, we employ Principal Component Analysis (PCA) (F.R.S., 1901) to retain the principal components explaining 95% of the variance in training hidden states. These dimensionally reduced hidden states are then used to fit GMMs approximating $P_{h_{\text{train}}}$ and $P_{h_{\text{test}}}$. Note that the number of principal components required to explain 95% of the variance in training hidden states serves as a measure of the vectorial complexity of a model's hidden states. In this context, these states inhabit a lower-dimensional hyperplane than that of the latent space. This measure, which we call Feature Vectorial Complexity (FVC), is reported in Table 4.

Observations and Analysis In Table 4, we observe that an increase in exposure bias is associated with lower F1 scores. To further investigate this trend, we compute the Spearman correlation coefficient between performance (AVG) and the corresponding EB values. This analysis was performed for all models and specifically for the SSLBS models. The resulting correlation coefficients are -0.59% (all models) and -89% (SSLBS models), indicating a robust negative association between these two variables, which validates our initial observation, highlighting the adverse effect of exposure bias on performance.

6.4. Effect of the Temperature parameter

We conducted experiments on ACE05-E+ varying the temperature parameter α in the range {0.1, 0.5, 1, 2, 5, 10} to study its impact on performance. As shown in Figure 5, the model with an intermediate temperature (α = 1) achieved the highest performance, indicating better training stability and model confidence calibration.

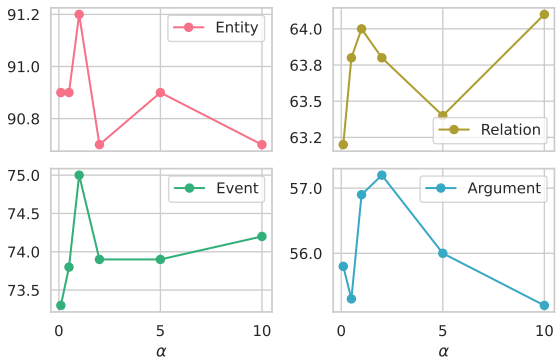

Figure 5: Effect of the temperature on performance - SSLBS - ACE05-E+ dataset.

We also experimented with annealed temperatures, starting from $\alpha = 1$ or $\alpha = 10$ and either decreasing linearly or exponentially towards $\alpha = 0.1$, but it did not yield any performance improvement.

7. Related Work

Many works addressed the entity recognition (ER) task separately (Zhou and Su, 2002; Tjong Kim Sang and De Meulder, 2003), others addressed the relation extraction (RE) task separately (Zelenko et al., 2002; Kambhatla, 2004), and others addressed both entity and relation extraction (ERE) tasks jointly (Chan and Roth, 2011; Zheng et al., 2017). Recent works address the four tasks; entity, relation, trigger, and argument extraction jointly (Luan et al., 2019; Wadden et al., 2019; Lin et al., 2020; Zhang and Ji, 2021; Nguyen et al., 2022b).

Seq2Seq Models Some works proposed Sequence-to-Sequence architectures for ERE. While Miwa and Bansal (2016) used an encoder-decoder architecture with attention, they relied on expensive trees. In contrast, Zheng et al. (2017) reformulated ERE as a single sequence labeling task but did not handle overlapping relations

effectively. To our knowledge, we are the first to recast the four tasks as a sequence labeling problem trained jointly.

Exposure Bias Solutions Several works addressed the issue of exposure bias in Seq2Seq models for various NLP applications, including ER, summarization, translation, and parsing. Solutions include reinforcement learning models (Ranzato et al., 2016), beam search training schemes with sequence-level cost functions (Wiseman and Rush, 2016), and differentiable relaxations of beam search procedures (Goyal et al., 2018). These methods have been applied to tasks such as NER

Our work Both our work and that of You et al. (2018a) use linearization to transform the structure of a graph into a sequential representation, enabling processing by autoregressive models. However, while You et al. (2018a) explicitly models the generative process of graph generation, our focus lies on predicting graph-related tasks.

Our continuous beam search procedure is similar to that of Goyal et al. (2018). Differently from them, we integrate four tasks into the procedure, we optimize an adapted loss to our task, and we use a straightforward RNN recurrence that implicitly integrates contributions from other beam elements to compute the next ones.

8. Conclusion

In this work, we present a novel joint information extraction model with a differentiable beam search. Our model optimizes two systems together: one for identifying entities and event triggers and the other for generating the labeled graph, which is recast as a sequence labeling problem. We conduct experiments that demonstrate the effectiveness of aligning the training and inference procedures.

9. Limitations

Our work has two main limitations. The first one is the order we chose for the graph linearization, specifically the lexicographic order. In fact, such an order is arbitrary and does not account for any language-related pattern. Forcing such an order makes it difficult for the model to infer the correct instance type patterns. In future work, we aim to make the model learn such an order itself.

The second limitation is related to vanishing gradients when the temperature parameter α decreases. This parameter is used to control the level of confidence in the model's predictions. This creates a trade-off between the stability and the accuracy of the predictions.

Acknowledgements

This work is partially supported by a public grantoverseen by the French National Research Agency (ANR) as part of the program Investissements d'Avenir (ANR-10-LABX-0083). This work was granted access to the HPC/AI resources of [CINES/IDRIS/TGCC] under the allocation 2023AD011013752R1 made by GENCI.

Bibliographical References

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for Arabic language understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1171–1179, Cambridge, MA, USA. MIT Press.

BSI. 1973a. *Natural Fibre Twines*, 3rd edition. British Standards Institution, London. BS 2570.

BSI. 1973b. Natural fibre twines. BS 2570, British Standards Institution, London. 3rd. edn.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

A. Castor and L. E. Pollux. 1992. The use of user modelling to guide inference and learning. *Applied Intelligence*, 2(1):37–53.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 551–560, Portland, Oregon, USA. Association for Computational Linguistics.

J.L. Chercheur. 1994. *Case-Based Reasoning*, 2nd edition. Morgan Kaufman Publishers, San Mateo, CA.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

N. Chomsky. 1973. Conditions on transformations. In *A festschrift for Morris Halle*, New York. Holt, Rinehart & Winston.

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

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

Markus Eberts and Adrian Ulges. 2019. Span-based joint entity and relation extraction with transformer pre-training.

Niama El Khbir, Nadi Tomeh, and Thierry Charnois. 2022. ArabIE: Joint entity, relation and event extraction for Arabic. In *Proceedings of the The Seventh Arabic Natural Language Processing Workshop (WANLP)*, pages 331–345, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Hao Fei, Shengqiong Wu, Jingye Li, Bobo Li, Fei Li, Libo Qin, Meishan Zhang, Min Zhang, and Tat-Seng Chua. 2022. LasUIE: Unifying information extraction with latent adaptive structure-aware generative language model. In *Advances in Neural Information Processing Systems*.

Karl Pearson F.R.S. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. 2017. Differentiable scheduled sampling for credit assignment. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 366–371, Vancouver, Canada. Association for Computational Linguistics.

Kartik Goyal, Graham Neubig, Chris Dyer, and Taylor Berg-Kirkpatrick. 2018. A continuous relaxation of beam search for end-to-end training of neural sequence models. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Paul Gerhard Hoel. 1971a. *Elementary Statistics*, 3rd edition. Wiley series in probability and mathematical statistics. Wiley, New York, Chichester. ISBN 0 471 40300.

Paul Gerhard Hoel. 1971b. *Elementary Statistics*, 3rd edition, Wiley series in probability and mathematical statistics, pages 19–33. Wiley, New York, Chichester. ISBN 0 471 40300.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. REBEL: Relation extraction by end-to-end language generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2370–2381, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*.

Otto Jespersen. 1922. *Language: Its Nature, Development, and Origin*. Allen and Unwin.

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 178–181, Barcelona, Spain. Association for Computational Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for 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, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Bo Li, Dingyao Yu, Wei Ye, Jinglei Zhang, and Shikun Zhang. 2022. Sequence generation with label augmentation for relation extraction.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of The 58th Annual Meeting of the Association for Computational Linguistics*.

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.

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. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*.

Meryem M'hamdi, Marjorie Freedman, and Jonathan May. 2019. Contextualized cross-lingual event trigger extraction with minimal resources. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 656–665, Hong Kong, China. Association for Computational Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany. Association for Computational Linguistics.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. 2017. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5949–5958, Red Hook, NY, USA. Curran Associates Inc.

Dat Quoc Nguyen and Karin Verspoor. 2019. End-to-end neural relation extraction using deep biaffine attention. In *Lecture Notes in Computer Science*, pages 729–738. Springer International Publishing.

Minh Van Nguyen, Viet Dac Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022a. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022b. Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9349–9360, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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. In *International Conference on Learning Representations*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Douglas A. Reynolds. 2009. Gaussian mixture models. In *Encyclopedia of Biometrics*.

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.

Charles Joseph Singer, E. J. Holmyard, and A. R. Hall, editors. 1954–58. *A history of technology*. Oxford University Press, London. 5 vol.

Jannik Strötgen and Michael Gertz. 2012. Temporal tagging on different domains: Challenges, strategies, and gold standards. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3746–3753, Istanbul, Turkey. European Language Resource Association (ELRA).

S. Superman, B. Batman, C. Catwoman, and S. Spiderman. 2000. *Superheroes experiences with books*, 20th edition. The Phantom Editors Associates, Gotham City.

Bruno Taillé, Vincent Guigue, Geoffrey Scoutheeten, and Patrick Gallinari. 2020. Let's Stop Incorrect Comparisons in End-to-end Relation Extraction! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3689–3701, Online. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Che-Ping Tsai and Hung-yi Lee. 2020. Order-free learning alleviating exposure bias in multi-label classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:6038–6045.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2773–2781, Cambridge, MA, USA. MIT Press.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. 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. Association for Computational Linguistics.

Jue Wang and Wei Lu. 2020. 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. Association for Computational Linguistics.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1296–1306, Austin, Texas. Association for Computational Linguistics.

Pengcheng Yang, Fuli Luo, Shuming Ma, Junyang Lin, and Xu Sun. 2019. A deep reinforced sequence-to-set model for multi-label classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5252–5258, Florence, Italy. Association for Computational Linguistics.

Deming Ye, Yankai Lin, Peng Li, and Maosong Sun. 2022. Packed levitated marker for entity and relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4904–4917, Dublin, Ireland. Association for Computational Linguistics.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018a. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pages 5708–5717. PMLR.

Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec. 2018b. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 5694–5703. PMLR.

Urchade Zaratiana, Nadi Tomeh, Pierre Holat, and Thierry Charnois. 2024. An autoregressive text-to-graph framework for joint entity and relation extraction.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 71–78. Association for Computational Linguistics.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1730–1740, Copenhagen, Denmark. Association for Computational Linguistics.

Ranran Haoran Zhang, Qianying Liu, Aysa Xuemo Fan, Heng Ji, Daojian Zeng, Fei Cheng, Daisuke Kawahara, and Sadao Kurohashi. 2020. Minimize exposure bias of Seq2Seq models in joint entity and relation extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 236–246, Online. Association for Computational Linguistics.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1227–1236, Vancouver, Canada. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. 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. Association for Computational Linguistics.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 473–480, USA. Association for Computational Linguistics.

10. Language Resource References

Walker, C. and Linguistic Data Consortium. 2005. *ACE 2005 Multilingual Training Corpus*. Linguistic Data Consortium, LDC corpora, ISLRN 458-031-085-383-4.