An Empirical Study of Representation, Training and Decoding for Span-based Named Entity Recognition

Anonymous ACL submission

Abstract

Named Entity Recognition (NER) is an important task in Natural Language Processing with application in many domains. While the dominant paradigm of NER is sequence labelling, span-based approaches have become very popular in recent times, but are less well understood. In this work, we study different aspects of span-based NER, namely the span representation, learning strategy, and decoding algorithms to avoid span overlap. We also propose an exact algorithm that efficiently finds the set of non-overlapping spans that maximize a global score, given a list of candidate spans. We perform our study on three benchmarks NER datasets from different domains. The code and supporting files for the experiments will be made publicly available.

1 Introduction

Named Entity Recognition (NER) is an important task in natural language processing whose goal is to identify and extract named entities such as person, organization, location from texts. NER systems are typically designed as sequence labelling, i.e., token-level prediction utilizing the BIO scheme (Lample et al., 2016; Huang et al., 2015). While traditional approaches use hand-crafted features along with classical Machine Learning algorithms such as SVMs or decision trees (Zhou and Su, 2002; Carreras et al., 2002; Li et al., 2004), recent deep learning models learn features directly from the data using for example bi-directional LSTMs (Hochreiter and Schmidhuber, 1997) or more recently pre-trained language models such as BERT (Devlin et al., 2019).

Recently, span-based NER has gained in popularity. Unlike sequence tagging which operates at the token level, span-based NER operates directly at the span level. The main idea is to enumerate all possible spans (contiguous sequence of tokens) of an input text and predict their identity using neural networks (Lee et al., 2017). One of the major advantages of the span-based NER is that it can learn a rich representation of the span instead of only learning the representation of each token. In addition, a recent study by Fu et al. (2021) reveals that span-based NERs are better in a context with more OOV words and Li et al. (2021) showed that span-based NERs are much better than sequence labelling in settings with unlabelled entities.

However, unlike sequence labelling, unconstrained span-based approaches tend to produce overlapping entities, which is undesirable for flat, non-overlapping NER tasks. To avoid overlap in span-based NER, two main approaches have been adopted in the literature. The first is the Semi-Markov conditional random field (Sarawagi and Cohen, 2005; Kong et al., 2016; Sato et al., 2017) that trains a globally normalized model and then uses a Viterbi algorithm to produce the optimal segmentation without span overlap. The second algorithm is the one employed by Li et al. (2021) for locally normalized span-based NER; it first eliminates all non-entity spans and deals with the overlap conflict by keeping the span with the highest prediction probability while eliminating the others. In this work, we call this approach "greedy decoding".

In this paper, we analyze and compare different decoding algorithms for the span-based NER. In addition to Semi-CRF decoding and greedy decoding, we propose an exact version of greedy decoding called "global-aware span selection" which mitigates the myopic bias of greedy decoding by returning a set of non-overlapping spans that maximize a global score, by formulating the problem as finding the Maximum Weight Independent Set (MWIS) (Hsiao et al., 1992; Pal and Bhattacharjee, 1996) of the span overlap graph. Furthermore, we also explored different types of span representation including max-pooling, convolution and endpoints (representing span by its extreme tokens).

In summary our contributions are as follow:

- We investigate different span representations for span-base NER when using pretrained transformer models.
- We propose an exact decoding algorithm to eliminate span overlap on locally trained span-based models that overcomes the myopic bias of the greedy approach (Li et al., 2021).
- Our study reveals that Semi-CRFs are competitive when training data is low but local span-based approaches are superior when many train data is available.
- Our empirical study finds that sequence labelling models provide a strong result when the number of entity types is few while span-based approaches are better when the number of entity types is large.

We will make code for models and experiments publicly available to facilitate future span-based NER research.

2 Span representation

The goal of the span representation is to represent each span, i.e., a contiguous sequence of tokens, of an input text in an embedding vector for span prediction tasks. In this paper, we denote $h_i \in \mathbb{R}^{d_h}$ the representation of the word at the position i and $\boldsymbol{s}_{ij} \in \mathbb{R}^{d_s}$ the representation of the span from index i to j with the width k; here d_h, $d_s \in \mathbb{N}^+$ are respectively the embedding size for word and span representations. The token representations are computed using BERT-based model (Devlin et al., 2019; Liu et al., 2019). However, since BERT-based tokenization divides the input words into subwords, we take the first subword to represent the whole words, which has proven to be very competitive for several token classification tasks (Beltagy et al., 2019). In the following, we present different types of approaches to represent spans.

Endpoints Endpoints span representation consist in representing spans using the representation of the tokens of its right and left extremities among with span width feature. Specifically, the span representation $\boldsymbol{s}_{ij}$ is computed as:

$$\boldsymbol{s}_{ij} = [\boldsymbol{h}_i; \boldsymbol{h}_j; \boldsymbol{w}_k] \qquad (1)$$

Span representation	Num params.
Endpoints	$(2d_h + d_k)C$
Maxpool	d_hC
Convolution	$\frac{1}{2}d_h^2K(K+1) + d_hC$
Convolution (shared)	$d_h^2K + d_hC$
FirstToken	$d_h^2K + d_hC$

Table 1: Number of parameters for different representation, without including the word representation layer which is the same for any approach. d_h, K and C are respectively the word embedding size, the maximum span width and the number of classes. Blue terms are parameters for computing span representations and Red terms denote number of parameters for the final layer.

In the above equation, $\boldsymbol{w}_k$ is a learned vector of span width k and $[;]$ denotes the concatenation operation. This representation has been widely used in previous works for span prediction tasks such as NER and coreference resolution (Lee et al., 2017; Luan et al., 2019; Zhong and Chen, 2021).

Max-pooling Since spans consist of a contiguous segment of tokens, pooling operations are a fairly natural way to compute span representations. In this context, we compute the representation of a span by applying an element-wise max-pooling operation to all tokens representing the span. Formally, the span representation of a span (i, j) is computed as follows:

$$\boldsymbol{s}_{ij} = \texttt{MAX}([\boldsymbol{h}_i; \boldsymbol{h}_{i+1}; \ldots; \boldsymbol{h}_j]) \qquad (2)$$

Where MAX is the element-wise max pooling operation. This maxpool span representation has been previously used by Eberts and Ulges (2020) for joint entity and relation extraction.

Convolution Instead of simply applying the pooling operation to the token spans, we explored the span representation by aggregating the tokens using learned filters via 1D convolution. Specifically, the representations of all spans of size k are computed using a 1D convolution of kernel size k over the token representations. An important consideration, however, is to keep the number of parameters linear with respect to the maximum span width, so we share the convolution weights across the different span widths.

$$\boldsymbol{s}_{ij} = \texttt{Conv1D}_k([\boldsymbol{h}_i; \boldsymbol{h}_{i+1}; \ldots; \boldsymbol{h}_j]) \qquad (3)$$

In the above equation, $\texttt{Conv1D}_k$ is a convolution filter of size k where k is the width of the span from i to j. Lei et al. (2021) used this convolutional approach to represent spans for neural keyphrase extraction.

FirstToken Here we compute the span representations only using the representation of the start token along with span width information:

$$\boldsymbol{s}_{ij} = \boldsymbol{W}^{(k)}\boldsymbol{h}_i \tag{4}$$

$\boldsymbol{W}^{(k)} \in \mathbb{R}^{d_h \times d_h}$ is the weight matrix associated to the span width k. Thus, for instance, the representation of the span (0, 0) and (3, 5) are respectivelly computed as $s_{00} = \boldsymbol{h}_0\boldsymbol{W}^{(1)}$ and $s_{36} = \boldsymbol{h}_3\boldsymbol{W}^{(4)}$. Note that the computation of the representation of all spans for this approach can be done in parallel and in a single line of code using, for example, the einsum operation. This representation was inspired by the synthetic attention from Tay et al. (2021), where the authors predict attention maps without pairwise interaction between tokens (resulting in linear attention).

Number of parameters The number of parameters required for each span representation is shown in Table 1. The endpoints and maxpool representations do not involve any parameters for the computation of the span representation but only for the final prediction. We can also observe that convolution with a shared filter considerably reduces the number of parameters.

3 Model

After representing the spans, the next step is to compute the prediction score or logits for each span. We use an affine transformation to compute the logits $\boldsymbol{y}_{ij} \in \mathbb{R}^C$ for a given span representation $\boldsymbol{s}_{ij}$:

$$\boldsymbol{y}_{ij} = \boldsymbol{W}^{(f)}\texttt{ReLU}(\boldsymbol{s}_{ij}) + \boldsymbol{b}_f \tag{5}$$

Here, $\boldsymbol{W}^{(f)} \in \mathbb{R}^{d_s \times C}$ is the final weight matrix, $\boldsymbol{b}_f \in \mathbb{R}^C$ is the bias vector, and `ReLU` is the activation function.

We denote by $\phi(i, j, y)$ the unnormalized score of the label y for the span (i, j). For training our models, we adopted two different approaches: a locally normalized approach and a globally normalized approach (or "Semi-CRF training") described in the following subsections.

3.1 Locally normalized model

In this configuration, we compute the prediction probability of each span independently using the softmax function:

$$p(y|i,j) = \frac{\exp \phi(i,j,y)}{\sum_{y'} \exp \phi(i,j,y')} \tag{6}$$

During training, we minimize the negative log-likelihood of the gold label spans:

$$\mathcal{L} = -\sum_{l=1}^{|Y|} \log p(y_l|i_l, j_l) \tag{7}$$

Where $|Y|$ is the total number of spans batch including all non-entity. Note that we did not employ any negative sampling on non-entity spans like (Li et al., 2021) since we assume that our training data are well annotated.

Furthermore, for this approach, some constrained decoding is applied during inference for solving overlap conflicts. For that, we used a two-step decoding. The first step consist in filtering the spans that has been predicted as "non-entity" by the model, i.e dropping span (i, j) when $p_\emptyset = \max_y p(y|i,j)$ with $p_\emptyset$ the probability of the span being non-entity. After this filtering step, the second step is to solve overlapping conflicts with greedy or global decoding which we detail in the following.

Greedy decoding Given the candidate spans provided by the filtering step, this greedy algorithm iteratively chose the max-probability entity span not overlapping any previously chosen entity span. This algorithm has a complexity of $O(n \log n)$. However, one potential issue of this algorithm is that it suffers from myopic bias. This decoding has been previously employed by (Li et al., 2021).

Global decoding The motivation for global decoding is that the greedy approach is likely to suffer from a myopic bias, i.e., a span is selected without considering the future decision. Thus, the goal of this global decoding is to select the best set of spans that maximizes a certain global score, thus neutralizing the myopic bias of the greedy algorithm. In this work, following (Li et al., 2021; Fu et al., 2021), we use probability to represent the score of spans.

$$s^* = \arg\max_{s \in \mathcal{S}} \sum_{i=1}^{|s|} \max_y p(y|s_i) \tag{8}$$

Where $\mathcal{S}$ contains the list of all possible candidates which are maximal and non-overlapping subsets of the spans from the filtering step. Here, the term "maximal" means that adding another span to the set would break the non-overlap constraint. To solve this optimization problem, we conceptualize it as finding the Maximum Weight Independent Set (MWIS) from the overlapping entity graph $\mathcal{G}$, represented by its adjacency matrix $\boldsymbol{A}$, defined as follows:

$$\boldsymbol{A}[x, y] = \begin{cases} 1, & \text{if } \texttt{has\_overlap}(x, y) \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

Where $\texttt{has\_overlap}(x, y)$ is a function that returns True if the spans x and y are overlapped and not equal, and the weight of each node correspond to their probability value provided by the model. Finally, the solution to equation 8 is provided by MWIS of the graph $\mathcal{G}$. We note that an "independent set" is a subset of node($\mathcal{G}$) without adjacent node (i.e a set of non-overlapping spans). For general graph, computing the MWIS is NP-Hard but since our graph is an interval graph (spans can be considered as intervals), computing the solution has a complexity of $O(S \log S)$ or $O(S)$ if the spans are sorted by their left endpoint (Hsiao et al., 1992).

We also consider the use of the average probability as a global score, that is by normalizing the original global score by the number of spans:

$$s^* = \arg\max_{s \in \mathcal{S}} \frac{1}{|s|} \sum_{i=1}^{|s|} \max_y p(y|s_i) \quad (10)$$

This decoding requires enumerating all possible candidates $\mathcal{S}$ which is NP-Hard, with a complexity of $O(3^{S/3})$ (Johnson et al., 1988; Raman et al., 2007). However, since the number of entities is limited for a well-trained model, the solution can be obtained very quickly. Empirically, we found that the number entity is approximately $0.15 \times L$ where L is the sentence length.

3.2 Globally normalized model

Rather than maximizing the likelihood for each span, this approach instead maximizes the probability gold segmentation Y of the input sequence against all possible segmentation using Semi-Markov CRF (Sarawagi and Cohen, 2005), which is an estimator of the form:

$$p(Y|x) = \frac{\exp \sum_l^{|Y|} \phi(i_l, j_l, y_l) + T_{y_l, y_{l-1}}}{Z(x)} \quad (11)$$

Decoding algorithm	Time complexity
CRF	$O(L\|Y\|^2)$
Semi-CRF	$O(LK\|Y\|^2)$
Greedy decoding	$O(S \log S)$
Global sum	$O(S \log S)$
Global mean	$O(3^{S/3})$

Table 2: This table reports the complexity of the different decoding algorithms. L is the input length, K the maximum segment width, $|Y|$ the number of classes and S the number of spans after filtering non-entities, which is approximately equal to $0.15 \times L$ empirically.

In the equation above, Y is the gold segmentation and $|Y|$ is it's size, and following (Sarawagi and Cohen, 2005), we assume that segments have strictly positive lengths, adjacent segments touch and we assume that non-entity spans have unit length. For instance, a segmentation of the sentence "Michael Jordan eats an apple ." would be *Y=[(0, 1, PER), (2, 2, O), (3, 3, O), (4, 4, O), (5, 5, O)]*. The terms $\phi(i_l, j_l, y_l)$ and $T_{y_l, y_{l-1}}$ are respectively the emission score and the transition score. Here, we assume a Markov dependence of order 1 between the labels. Finally, the $Z(x)$ term in the denominator is a normalization factor (or "partition function") which is computed as follows:

$$Z(x) = \sum_{Y' \in \mathcal{Y}} \exp \sum_{l'}^{|Y'|} \phi(i_{l'}, j_{l'}, y_{l'}) + T_{y_{l'}, y_{l'-1}} \quad (12)$$

Where $\mathcal{Y}$ contains all possible segmentations of the input sequence. This normalization term can be computed in polynomial time using dynamic programming (Sarawagi and Cohen, 2005). During training, the objective is to maximize the log-likelihood $\log(Y|x)$ on the training set.

Viterbi decoding For decoding Semi-CRF model, we the segmentation version of the viterbi algorithm from Sarawagi and Cohen (2005) which produce the optimal segmentation of the input sequence.

4 Experiments setup

4.1 Datasets

We evaluated our model on three benchmark datasets for Named Entity Recognition: Conll-2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes 5.0 (Weischedel et al., 2013) and TDM dataset (Hou et al., 2021). Conll-2003 is a

Model	Span Representation											
	Convolution			*Endpoints*			*Maxpool*			*FirstToken*		
	P	R	F	P	R	F	P	R	F	P	R	F
Conll-2003												
Local	90.48	90.39	90.44	91.05	90.25	90.65	91.51	90.74	91.12	90.56	89.72	90.14
+ Greedy	91.51	90.13	90.81	91.46	89.76	90.60	92.15	90.37	**91.25**	90.74	89.58	**90.16**
+ Global sum	91.47	90.11	90.79	91.42	89.74	90.57	92.05	90.34	91.18	90.74	89.58	**90.16**
+ Global mean	91,54	90.14	**90.84**	91,60	89.83	**90.70**	92.15	90.37	**91.25**	90.74	89.58	**90.16**
Semi-CRF	89.73	89.51	89.62	89.37	89.25	89.31	89.11	88.58	88.85	89.18	89.16	89.17
OntoNotes 5.0												
Local	88.40	89.18	88.79	88.87	89.36	89.12	88.55	89.34	88.95	87.89	88.99	88.44
+ Greedy	89.19	88.84	**89.02**	89.50	88.95	89.22	89.49	88.95	**89.22**	88.70	88.44	88.57
+ Global sum	89.16	88.84	89.00	89.51	88.98	**89.24**	89.44	88.95	89.20	88.73	88.49	**88.61**
+ Global mean	89.19	88.81	89.00	89.50	88.95	89.22	89.52	88.93	**89.22**	88.71	88.44	88.58
Semi-CRF	87.69	87.46	87.58	87.33	88.44	87.88	86.89	88.53	87.70	87.82	87.85	87.83
TDM												
Local	76.68	63.50	69.47	72.89	68.93	70.85	71.98	70.14	71.05	65.85	65.16	65.50
+ Greedy	77.16	63.20	69.49	75.37	68.33	71.68	73.42	69.98	**71.66**	66.88	64.25	65.54
+ Global sum	77.16	63.20	69.49	75.37	68.33	71.68	73.42	69.98	**71.66**	66.88	64.25	65.54
+ Global mean	77.16	63.20	69.49	75.46	68.17	71.63	73.38	69.83	71.56	66.88	64.25	65.54
Semi-CRF	67.89	72.40	**70.07**	69.25	75.41	**72.20**	69.63	68.48	69.04	68.78	70.44	**69.60**

Table 3: This table reports the main results of our study. It shows the performance along different settings including the datasets, the trainings, decodings and span representations. **Bold** numbers indicate the best model/decoding for a fixed representation and underlined numbers indicate the best representation for a fixed model/decoding.

dataset from the news domain that was designed for extracting entities such as person, location and organisation from texts. OntoNotes 5.0 is a large corpus comprising various genres of text including newswire, broadcast news, broadcast conversation, magazine, web data and telephone conversation. It contains in total 18 different entity types such as Person, organization, location, product or date. TDM is a NER dataset that was recently published and it was designed for extracting Tasks, Datasets, and Metrics entities from Natural Language Processing papers. In our study, we set the maximum width of the spans to 6 tokens and reject data instances that contain entities longer than this maximum value.

4.2 Evaluation metrics

We evaluate our models on the test splits of the corresponding datasets. Our evaluation is based on the exact match between true and gold entities by discarding non-entity spans. We report the micro-averaged precision, recall and F1.

4.3 Implementation details

Backbone models As base models, we used RoBERTa-base (Liu et al., 2019) for models trained on Conll-2003 and OntoNotes 5.0 because they come from general domains and we employed SciBERT (Beltagy et al., 2019) for models trained on TDM, which is a scientific NER data set.

Dataset	P	R	F
Conll-2003	91.36	90.63	90.99
OntoNotes 5.0	88.10	89.38	88.74
TDM	66.35	74.36	70.13

Table 4: Performance for the baseline sequence labelling approach, a BERT-CRF tagger.

Baseline We compare the span-based approaches to a sequence labelling BERT-CRF, same as the NER model used by (Beltagy et al., 2019). Specifically, this model first encodes an input sequence using a pretrained transformer model and the final token representations are linearly projected to compute the logits. A Conditional Random Field (Lafferty et al., 2001) layer is also added to the output to ensure valid BIO tagging during decoding by constraining the transition matrix, which facilitates the evaluation.

Hyperparameters All models were trained for up to 25 epochs using Adam (Kingma and Ba, 2017) as the optimizer with a learning rate of 1e-5. We opted for a batch size of 10, except for the Semi-CRF models trained on OntoNotes, where our GPU ran out of memory, because this dataset has 18 en-

Dataset	Model	#Examples						
		100	250	500	1000	2500	5000	All†
Conll-2003	CRF	66.68	**78.84**	**81.42**	**85.07**	**87.67**	88.11	**90.99**
	Local	65.08	71.74	77.23	82.81	85.55	87.76	90.65
	+ Greedy	68.44	74.02	78.70	83.63	86.50	88.31	90.60
	+ Global sum	68.65	74.29	78.59	83.58	86.50	**88.32**	90.57
	+ Global mean	67.37	73.45	78.64	83.70	86.40	88.20	90.57
	Semi-CRF	**70.28**	73.84	79.00	83.49	87.08	87.84	89.31
OntoNotes 5.0	CRF	61.93	68.14	72.66	77.2	80.23	81.84	88.74
	Local	60.34	67.79	73.53	76.92	80.09	81.92	89.12
	+ Greedy	62.38	70.14	74.98	77.50	**81.39**	**82.65**	89.22
	+ Global sum	**63.35**	**70.25**	**75.03**	77.54	81.28	82.62	**89.24**
	+ Global mean	61.73	69.73	74.79	77.44	81.34	82.55	89.22
	Semi-CRF	63.08	70.09	73.96	**78.12**	80.24	81.86	87.88
TDM	CRF	**62.28**	**67.78**	**69.92**				70.12
	Local	57.25	63.15	67.65				70.85
	+ Greedy	58.46	65.28	67.84				71.68
	+ Global sum	60.55	65.18	67.62				71.68
	+ Global mean	57.54	61.84	67.46				71.63
	Semi-CRF	60.49	65.06	66.52				**72.20**

Table 5: Performance (F1-score) across all datasets and different training set sizes. † The train size of Conll-2003, OntoNotes 5.0 and TDM are respectively 11110, 48788 and 980.

tity types, which results in a large transition matrix, and so we used a batch size of 5 and added a 2-step gradient accumulation. We further used early stopping with a patience of 5 (on the F1 score) and keep the best model on the validation set for testing. We implement our model with pytorch (Paszke et al., 2019). The pre-trained transformer models were loaded from the HuggingFace's Transformers (Wolf et al., 2020). We employed AllenNLP (Gardner et al., 2018) for data preprocessing and the seqeval library (Nakayama, 2018) for evaluating the baseline sequence labelling model. Furthermore, Our Semi-CRF implementation is based on pytorch-struct (Rush, 2020). All models were trained using a single V100 GPU. For this project, we used approximately 200 GPU hours.

5 Results

5.1 Span representation

We analyze the performance of the span representations on both the local model and the Semi-CRF model, as shown in the following table 3.

Local model Using the local model without decoding, we notice that the best results on the task are obtained by the Maxpool and Endpoint representations. The Maxpool representation shows the best result on the Conll-2003 and TDM datasets and the second best result on OntoNotes 5.0; while the Endpoint representation scores the best result on OntoNotes 5.0 and the second best on the Conll-2003 and TDM datasets. In addition, the convolution representation performed competitively, but it is consistently inferior to Maxpool and Endpoints, which is somewhat surprising on account of it having a larger number of parameters. Finally, the FirstToken approach has the weakest performance on all datasets but can maintain reasonable performance while using only a single token representation to represent a span, demonstrating that the token representation of pre-trained transformer models is indeed capable of holding rich information.

Global model With the Semi-CRF, the Endpoints representation remains competitive by achieving the best result on two of the three datasets. However, unlike the local model, Maxpool gets the worst performance; it gets the lowest F1 score on the conll-2003 and TDM datasets, which is quite surprising. Furthermore, while FirstToken was the worst representation on the local model, we found that it can sometimes outperform other approaches in the Semi-CRF framework. Finally, the convolution approach has the best result on conll-2003

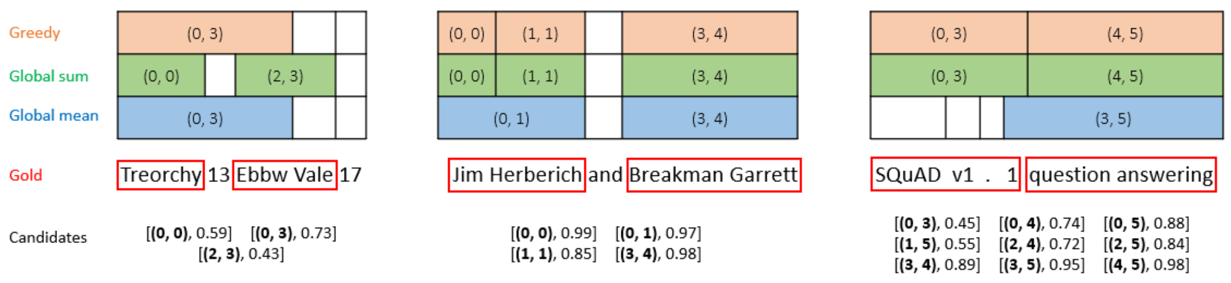

Figure 1: Shows how overlapping conflicts are handled by the different decoding algorithms. We only include overlaps involving at least three entities, because otherwise all decodings produce the same result.

and the second best on TDM, while performing the worst on OntoNotes 5.0.

Overall, we found that the representation performance is not consistent across the different training schemes, except for the Endpoints that were found to be competitive in both cases.

5.2 Decoding algorithm

We herein compare the different types of decoding algorithms to avoid overlapping in the span-based NER. Table 3 shows the performance results of the different decoding algorithms under different settings. For the local models, one of the first things we can notice is that applying the application of decoding, most of the time improve the performance. However, there is no significant difference between greedy and global decodings since the models are already well trained and thus, overlap filtering does not provide much difference in terms of quantitative results.

Qualitative analysis As we did not have a large difference in performance between the greedy decoding and the two global decodings, we performed a qualitative analysis to compare the three approaches. This study is presented in Figure 1, which shows the input text (truncated), the raw prediction with overlap, and the results after applying greedy decoding and the global decodings (mean and sum). We only include overlaps involving more than two spans, because when two spans overlap, all algorithms take the span with the highest probability.

We can see that the greedy approach always retrieves the most probable entity since it iteratively selects the best spans that do not overlap with previously selected spans. However, this algorithm tends to suffer from a myopic bias. Second, the global sum approach, which maximizes the sum of probabilities, tends to select as many spans as possible, which means that it favours shorter spans over longer ones. Also, global sum decoding has a slightly higher recall score most of the time than other decoding algorithms. Finally, global average decoding, which selects the set of spans that maximizes the average probability, tends to select the smallest number of spans, but the selected spans generally have a high probability. In general, this decoding tends to favour precision over recall score.

5.3 Few-shot performance

We conduct a study to compare the performance of each model in a few-shot scenario. The evaluation is performed on the test set of each dataset using from 100 to the full training dataset. For this study, we use the Endpoints representation for spans because it is the most widely used representation in the literature and it consistently achieves good performance across different training and decoding schemes. The results of the few-shot evaluation are presented in Table 5.

Semi-CRF is better than the local spans-based approach when overlap filtering is not performed. Moreover, using only 100 training data on Conll-2003, a Semi-CRF model can outperform a local model by up to 5 points in terms of F1 score. However, most of the time, the local approach performs better than Semi-CRF when the number of data is large (>2500).

In general, the local model with decoding, especially global mean decoding, outperforms Semi-CRF in both low and high data settings. Moreover, in a few-shot scenario using the local model, decoding can increase the F1 by more than 3 points. Furthermore, we also remark that the increase in performance by decoding is higher when a local model is training on a few datasets while the difference becomes less significant when the number of

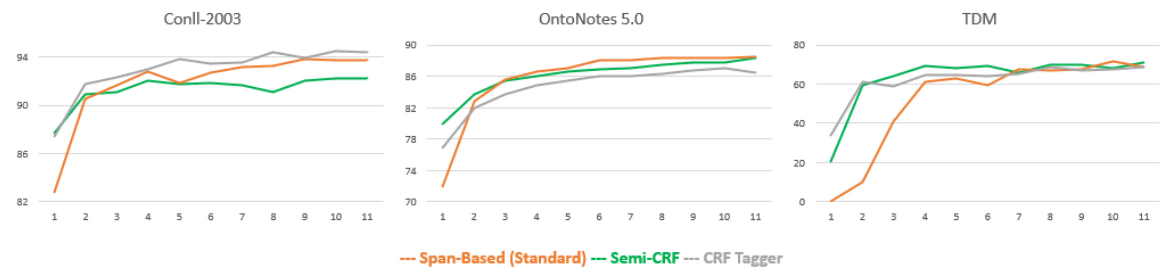

Figure 2: The learning curve for all the datasets and for different training scheme. The x-axis of the charts represents the learning epochs and the y-axis shows the F1-score on the development set.

training data is large.

We find that the baseline sequence labelling, BERT-CRF approach is indeed competitive. It most of the time obtains a better performance on Conll-2003 and TDM datasets across any dataset sizes. However, the span-based approach is better on the OntoNotes 5.0 dataset. This can be explained by the fact that OntoNotes 5.0 contains 18 entity types and, therefore, the labelling approach would require 54 labels since it uses a BIO scheme, which makes the task much more difficult.

5.4 Learning curve

In the following, we discuss the training speed of the different training schemes by analyzing the learning curve shown in Figure 2, which shows the evolution of the F1 score across training epochs. We observe that the Semi-CRF model can achieve better results with only a few learning steps compared to the local model. Thus, Semi-CRF models are not only more data-efficient but also require fewer learning steps. However, after a sufficient number of epochs, the local model generally outperforms the Semi-CRF model. In terms of computational speed, empirically, for the same number of training steps, it typically takes 1.5 to 2 times longer to train a Semi-CRF compared to the local span-based approach, as the computation of the Semi-CRF's partition function $\mathcal{Z}$ is computationally expensive.

6 Related Works

Different approaches for NER NER is an important task in Natural Language Processing and is used in many downstream information extraction applications. Usually, NER tasks are designed as sequence labelling (Chiu and Nichols, 2016; Huang et al., 2015; Ma and Hovy, 2016; Lample et al., 2016; Strubell et al., 2017; Rei, 2017; Akbik et al., 2018) where the goal is to predict BIO tags. Recently, different approaches have been proposed to perform NER tasks that go beyond traditional sequence labelling. One approach that has been widely adopted is the span-based approach (Liu et al., 2016; Luan et al., 2018, 2019; Fu et al., 2021; Li et al., 2021) where the prediction is done in the span level instead of entity level. Li et al. (2020) has also approached NER as a question answering task in which named entities are extracted by retrieving answer spans. In addition, recent work such as (Cui et al., 2021) considers NER as template filling by fine-tuning a BART (Lewis et al., 2019) encoder-decoder model.

Decodings For the spans-based approach, Semi-Markov has been used previously (Sarawagi and Cohen, 2005; Liu et al., 2016; Kong et al., 2016; Sato et al., 2017; Ye and Ling, 2018), however, their use with a BERT-type model has been little explored, something we did in this paper. The work of Fu et al. (2021) and Li et al. (2021) employed a heuristic decoding to avoid overlap for span-based NER. Their algorithm iteratively chooses the maximum probability entity span that does not overlap with a previously chosen entity span. In this paper, we have proposed an exact version of this algorithm.

7 Conclusion

In this paper, we present an empirical study of different aspects of span-based NER. We found that any design choice, such as span representation, training strategy, and decoding can have significant impacts on task performance. We also found that the performance of different approaches can vary depending on the size of the training dataset, the number of feature types in the dataset as well as the number of training steps.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text.

Xavier Carreras, Lluís Màrquez, and Lluís Padró. 2002. Named entity extraction using AdaBoost. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. Template-based named entity recognition using bart.

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

Markus Eberts and Adrian Ulges. 2020. Span-based joint entity and relation extraction with transformer pre-training. *ArXiv*, abs/1909.07755.

Jinlan Fu, Xuanjing Huang, and Pengfei Liu. 2021. SpanNER: Named entity re-/recognition as span prediction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7183–7195, Online. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Yufang Hou, Charles Jochim, Martin Gleize, Francesca Bonin, and Debasis Ganguly. 2021. TDMSci: A specialized corpus for scientific literature entity tagging of tasks datasets and metrics. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 707–714, Online. Association for Computational Linguistics.

Ju Yuan Hsiao, Chuan Yi Tang, and Ruay Shiung Chang. 1992. An efficient algorithm for finding a maximum weight 2-independent set on interval graphs. *Information Processing Letters*, 43(5):229–235.

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

David S Johnson, Mihalis Yannakakis, and Christos H Papadimitriou. 1988. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Lingpeng Kong, Chris Dyer, and Noah A. Smith. 2016. Segmental recurrent neural networks. *CoRR*, abs/1511.06018.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Yanfei Lei, Chunming Hu, Guanghui Ma, and Richong Zhang. 2021. Keyphrase extraction with incomplete annotated training data. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 26–34, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020. A unified mrc framework for named entity recognition.

Yangming Li, lemao liu, and Shuming Shi. 2021. Empirical analysis of unlabeled entity problem in named entity recognition. In *International Conference on Learning Representations*.

Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. 2004. Svm based learning system for information extraction. In *Deterministic and Statistical Methods in Machine Learning*.

Yijia Liu, Wanxiang Che, Jiang Guo, Bing Qin, and Ting Liu. 2016. Exploring segment representations for neural segmentation models. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2880–2886. AAAI Press.

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

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. 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.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Madhumangal Pal and GP Bhattacharjee. 1996. A sequential algorithm for finding a maximum weight k-independent set on interval graphs. *International Journal of Computer Mathematics*, 60(3-4):205–214.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Venkatesh Raman, Saket Saurabh, and Somnath Sikdar. 2007. Efficient exact algorithms through enumerating maximal independent sets and other techniques. *Theory of Computing Systems*, 41(3):563–587.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. *arXiv preprint arXiv:1704.07156*.

Alexander M. Rush. 2020. Torch-struct: Deep structured prediction library.

Sunita Sarawagi and William W Cohen. 2005. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press.

Motoki Sato, Hiroyuki Shindo, Ikuya Yamada, and Yuji Matsumoto. 2017. Segment-level neural conditional random fields for named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 97–102, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate sequence labeling with iterated dilated convolutions.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2021. Synthesizer: Rethinking self-attention in transformer models. *ArXiv*, abs/2005.00743.

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.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

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.

Zhixiu Ye and Zhen-Hua Ling. 2018. Hybrid semi-Markov CRF for neural sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–240, Melbourne, Australia. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.