Explicitly Modeling the Context for Chinese NER

Anonymous ACL submission

Abstract

Named entity recognition (NER) is the foundation of many natural language processing tasks. Current NER models have achieved promising results. But as pointed by several studies, they fail with a high ratio on generalization tests such as invariance test because they heavily rely on name information. So, we propose a context module to explicitly model the contextual information, and a trainable balance factor is designed to incorporate the result of context module. To learn this factor, we propose several tailored data augmentation strategies to generate synthetic labels for it. These approaches help the model learn whether it should focus on the context. Our method achieves on average 1.2% absolute improvement of F1 than BERT-CRF on three datasets. Moreover, our method performs on par with the best solutions who rely heavily on external features besides BERT. We also conduct invariance test to analyse the effect of the context information. The source code of our model and augmentation strategies will be available at Anonymous.url[1].

1 Introduction

Named Entity Recognition (NER) aims to extract word spans in natural language that mention real-world entities, and classify them into a predefined set of types like person, address, etc. It is the up-stream task of many NLP tasks like relationship extraction (Cao et al., 2019) and event extraction (Wadden et al., 2019).

NER models have achieved promising results on many benchmark datasets, but recent researches point out that current NER models are fragile and unreliable. Gui et al. (2021) find that the performance of NER models drops sharply when unseen phrases occur in the sample (drops around 10% and 20% respectively). Ribeiro et al. (2020) proposed a "CheckList" to reveal the generalization capacity of NLP models. Among the list, *Invariance Test* expects the model prediction to remain the same when facing with label-preserving perturbations to inputs on Quora Question Pairs task. In their tests, models perform poorly with a 10% failure rate.

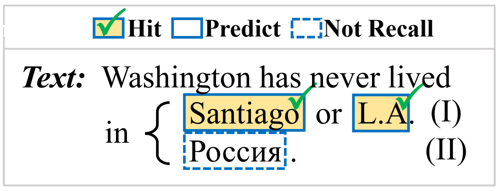

Figure 1: Invariance test: replacing a mention with another mention with the same type may lead to model failure.

Motivated by these work, we propose a "invariance test" for NER by replacing a mention with another mention with the same type, as demonstrated in Figure 1. We find BERT-based CRF (Devlin et al., 2018) model which achieves acceptable performance also fails on such small perturbation. Such behavior is further studied in Section 5.6, the failure rate reaches 19.0% and 8.1% for well-known CRF (Lafferty et al., 2001) solutions. It indicates that, some essential skills for NER are neglected by current models.

Current NER models are fragile because in many cases remembering the name, or exploiting biased cues in mentions is simpler than considering a long context, so that NER models tend to learn such shortcut (Peng et al., 2020; Field and Tsvetkov, 2019), which leading to a poor performance for entities not appearing in training data (Agarwal et al., 2020b). Here, "*name*" means the surface form of a text span, and "*context*" is the rest of the sentence. We human can perform well on NER because we are able to effectively distinguish the information from both the name and context. We can identify a company mention by a clear pattern in names like "ABC Co., Ltd.", or identify an ambiguous mention by looking at its context. Xu et al. (2017) also claim that contexts play an important role when it

[1] github.com/anonymous/anonymous

involves multi-sense words or phrases. Although such phenomenon has been discovered, to our best knowledge, few work has proposed to tackle it.

In this work, we propose to explicitly modeling the context information as an attempt to this problem. We adopt a span-based method that first detects possible spans and then classifies them into an entity type or the "None" type. In the span classification step, the model judges whether the span is an entity with two modules: A name module focuses on the information within the span, while a context module focuses on the information outside the span. Then, a combination module integrates them using a balance factor α to dynamically adjust their weight, and makes the final prediction. Note that the name module is an ordinary NER model can also look at the context, but we train it to focus on the name, and the context module can only look at the context. This is because if we separate the name and context of a span, then we need a heavy model to combine them. Our design has a ordinary module that looks at the whole sentence while can turn to context module is necessary.

However, existing datasets only annotate whether a span is an entity mention, but do not explain when context is more important to identify the mention. Without such supervision data, the proposed two-channel model will still lean to the name module. To solve this issue, we design tailored data augmentation strategies. For example, the "mention replacement (random)" strategy replaces an entity mention in a sentence with a random string. The context module is asked to predict the random string as an entity mention, the name module is asked to predict it as non-mention, and α leans to context. This encourages the model to discovery unseen entities. More details are discussed in Section 3.4.

We conduct experiment on several Chinese datasets, and leave other languages for further study. Experimental results show that our NC model performs on par or better than state-of-the-art NER models on several datasets, even though they incorporate heavy external resources. At the same time, our method significantly outperforms existing models on invariance test. Our main contributions are summarized as follows:

- We propose a two-module model called "NC-Model" with an context module to explicit model the context information. It performs significantly better on F1-Score than models using the same resource.
- We propose tailored data augmentation strategies to weakly supervise the model to learn when to focus on context information.
- We propose a "Invariance Test" to measure the generalization capacity of NER model on unseen names or contexts. Our proposed method performs significantly better on this test.

2 Related Work

Currently, NER models have achieved promising results on many benchmark datasets with various external resources. For example, the state-of-the-art models (Robert et al., 2021a) on Chinese NER (CNER) tasks dataset have achieved high F1-scores. Recent research pushes the limit by making use of a large lexicon and gazetteer (Zhang and Yang, 2018; Li et al., 2020), font features (Meng et al., 2019; Xuan et al., 2020), as well as a large corpus for pre-trained language models (Brown et al., 2020; Liang et al., 2020).

Existing solutions for NER tasks fall into two main categories: sequence labeling and span-based methods. Most recent work on NER are based on sequence labeling (Huang et al., 2015; Lample et al., 2016) to assign a label for each token. They focus on how to model a better text representation, such as character LSTM (Dong et al., 2016), character CNN (Ma and Hovy, 2016; Zhu et al., 2019) or Bert (Devlin et al., 2018). Current best CNER models are all using sequence labeling methods.

Besides sequence labeling, span-based (Eberts and Ulges, 2019; Zhong and Chen, 2021) solutions are also common in NER. This kind of method enumerates all spans in a text, filter them, and then assign a label to each span. The number of all spans in a sentence is usually large ($O(|s|^2)$). For this reason, the span-based solutions mentioned above drop spans longer than 6 (Tan et al., 2020; Xia et al., 2020) or 10 (Lee et al., 2017; Eberts and Ulges, 2019; Joshi et al., 2020) tokens at first. Our solution also utilize the span-based method for better modeling the context features.

Currently, there are researches that diagnose existing NER models and discuss the importance of name and context information for NER. Fu et al. (2020) systematically diagnosed current state-of-the-art NER models. They observed that the performance of existing models (including the latest ones) is largely affected by the extent to which test entities have been seen in the training set with the same label. Agarwal et al. (2020b) designed a model to classify mention purely depends on the context. The result indicated that only use context is not possible to achieve a high performance NER model.

The representations for context information attracts a lot of attention. Yaghoobzadeh and Schütze (2015) took the concatenation of the embeddings of the surrounding words of the mention as context representation. They pointed out that explicitly distinguishing name and context separately for modelling performed better. Lin et al. (2020a) prepared entity-triggers to solve with zero-shot entity strings. The triggers in context are able to mark the position of an entity in the sentence, to help the model effectively improve its generalization ability. However, the model requires extra annotation effort to annotate the trigger words of entity mentions. Lin et al. (2020b) claimed that there is a strong regularity between the names of same-labeled entities in the training and validation sets in the same dataset. They find that breaking the regularity (e.g., replacing the span with a random string) makes it harder for the model to predict correctly. Agarwal et al. (2020a) even replace entities with the ones in other languages. Other attempts on context information with max-pooling on Bert (Eberts and Ulges, 2019) or self-attention on Bi-LSTM (Xin et al., 2018), are also far behind existing CNER models mostly rely on names.

3 Name-and-Context Model

3.1 Overview

Given a pre-defined entity type set $\mathcal{E}$, the NER task aims to extract entities from a given sentence $x = (w_1, w_2, ..., w_n)$. The result is an entity set $\hat{E} = \{(i, j, t) \mid 0 \le i \le j < |x|\}, t \in \mathcal{E}$.

We want to explicitly model the context of a text span. Sequence labeling (Huang et al., 2015; Dong et al., 2016) method identify and type the mention simultaneously, which are not suitable for modeling spans. Therefore, we use the span-based method (Stratos, 2017; Tan et al., 2020) in this paper which first selects span candidates and then classifies them into pre-defined types $L = \mathcal{E} \cup \{\texttt{None}\}$, where None is the type for non-entity spans. A span $(w_i, ..., w_j)$ is denoted as $s = (i, j), s \in S$.

We name our model as **NC Model** (in short of "Name-and-Context Model"), as the entity type classification model has name module and context module. We also propose augmentation strategies for the classification step to encourage the model to lean on context modules under proper situations.

In the following sections, we introduce the span selection model, the span classification model, and finally the tailored data augmentation strategies in turn.

3.2 Span Selection

The span selection step filters all spans S in the text to obtain a subset S^f for classification. Most span-based methods (Lee et al., 2017; Zhong and Chen, 2021) enumerate all possible spans whose length are smaller than a threshold $L_{sp} \in \{6, 8, 10\}$. It is effective for English corpus as the length is measured at the word or sub-word level. For Chinese corpus with longer text spans at the character level, it is not enough. Millions of spans need to be judged, it takes too much time for training or inference. So that besides the simple length limit L_{sp}, we train a span scoring model $g(\cdot)$ for scoring each text span $g(s) \in [0, 1]$. The model classifies with both the boundary tokens of a span $s = (i, j)$:

$$\begin{aligned} (h_1, ..., h_n) &= \text{Encoder}(x) \\ g(s) &= Sigmoid(\text{FFN}(h_i \oplus h_j)) \end{aligned} \quad (245)$$

where h_i is the hidden vector for w_i from the Encoder, $\oplus$ denotes the concatenation operation.

Moreover, we utilize the *sentence segmentation* as another filtering rule. We remove spans which locate in more than one segments, so for each span candidate $s = (i, j)$, it needs to comply:

$$(w_i, w_{i+1}, ..., w_j) \cap \{\text{，}\ \text{；}\ \text{。}\ \text{？}\ \text{！}\ \ldots\} = \emptyset$$

which means we only generate spans between two adjacent delimiters. It reduces the number and average length of span candidates.

In a two-step NER approach, the recall of span classification is bounded by the span selection step. The training process needs *strong* negative spans (Eberts and Ulges, 2019), while the inferring needs span candidates that cover *more* entity mentions. At the same time, the number of span candidates is too large in longer sentences. It results in Out-Of-Memory issues during training with regular filtering methods (e.g., negative sampling by a certain proportion). There is a trade-off between the recall rate and span counts. After sorting with the span scores, We take the top $\theta_{\text{top-sp}}$ spans in training and evaluating.

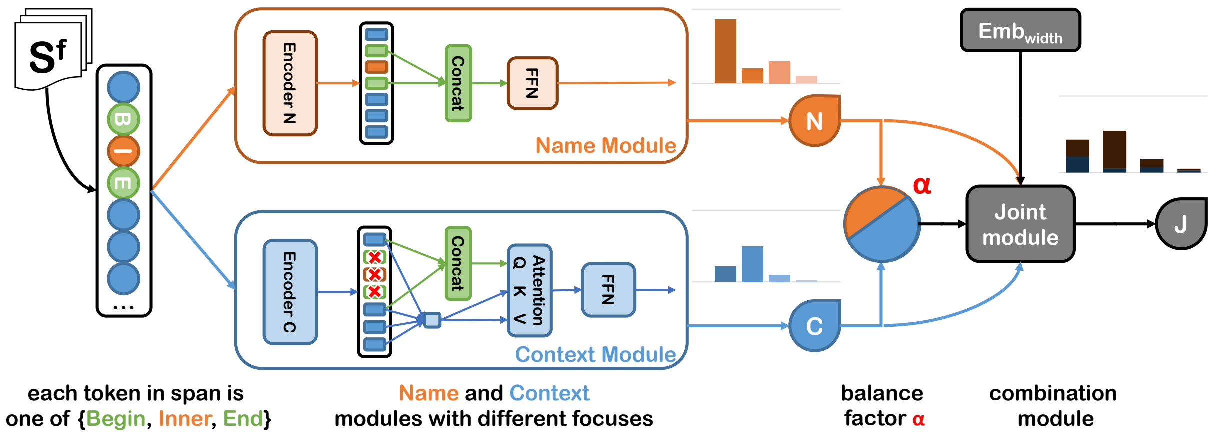

Figure 2: The framework for the span classification step in NC Model. The name module considers the information of the whole sentence, while the context module focus the context.

3.3 Span Classification

The span classification consists three modules: name, context, and combination modules. The name module focuses on the information inside the span, the context module focuses on the information outside the span, and the combination module balance these two modules. For a sentence $x = (w_1, ..., w_n)$ and a span $s = (w_i, w_{i+1}, ..., w_j)$, we introduce these three modules as follows.

The name module is demonstrated in the orange parts in Figure 2. We get the representation $h_{name}(s)$ of span s in sentence from an encoder using its head and tail word representations:

$$(h_1, ..., h_n) = \text{Encoder}(x)$$
$$h_{name}(s) = h_i \oplus h_j$$

where h_i is the hidden vector from the Encoder, $\oplus$ is the concatenation operation.

The context module is demonstrated in the blue parts in Figure 2. To prevent the shortcut learning from ignoring the context during the training process, we intentionally model the context separately.

x is fed into a bi-directional LSTM to get two hidden vectors for each token in forward and backward direction $(h_1, ..., h_n) = \text{BiLSTM}(x)$, where $h_i = [\overrightarrow{h_i}; \overleftarrow{h_i}]$ is the concatenation of hidden vectors from two directions. The context hidden vectors h'_{ctx} does not utilize any from the name information. Then we take an attention module based on $h'_{ctx}(s)$:

$$\begin{aligned} q &= h'_{ctx}(s) = [\overrightarrow{h_{i-1}}, \overleftarrow{h_{j+1}}] \\ K = V &= [\overrightarrow{h_1}, ..., \overrightarrow{h_{i-1}}, \overleftarrow{h_{j+1}}, ..., \overleftarrow{h_n}] \\ h_{ctx}(s) &= \text{Softmax}(q \cdot K^T / \sqrt{n_h}) \cdot V \end{aligned} \quad (1)$$

where n_h is the dimension of hidden vector. Notice we only attend on $\overrightarrow{h_k}$ if $k < i$ and $\overleftarrow{h_k}$ if $k > j$ so that $h_{ctx}(s)$ does not contain name information. In this way, we generate the sentence representations for all spans at the same time. Other ways to obtain context features, such as masking tokens in mention (Field and Tsvetkov, 2019) or inserting boundary tokens around the mention (Zhong and Chen, 2021), spend much more time generating sentence representations for every single span. The context module is obvious weaker than the name module, because it only covers contextual tokens. We do not care the performance of this module itself. What we focus on is whether the extra pure context information improves the NC Model.

It is worth mentioning that we do not strictly separate the context from the name module. In our model design, we leverage the pre-trained language model like BERT to offer semantic features for the name module. So the name module can peek at the context. We design the name module like this with the following concern: 1) The strictly decoupled name module plays the same role as a gazetteer or lexicon. There is no need to degrade the name module to achieve strict decoupling. 2) In the case where the name is confusing, the alpha value will lead the model to depend on the context module that does not contain tokens in the name. Even if both modules correctly focus on the same features in some cases, the model can still effectively answer with any alpha value.

The combination module combines h_{name} and h_{ctx} to make a final prediction. In addition, we utilize width embeddings emb_{width} for the span length feature. A balance factor $\alpha \in [0, 1]$ is introduced to balance between name and context modules. The final representation of the span is:

$$h_{nc}(s) = \alpha h_{name}(s) + (1-\alpha)h_{ctx}(s) \quad (2)$$

$$h(s) = h_{nc}(s) \oplus emb_{\text{width}}(|s|) \quad (3)$$

If α is close to 1, the final result depends more on the name module, otherwise more on the context module. In our primary experiments, we tried several methods to compute α, such as feed forward network and bi-affine. We find bi-affine performs slightly better. So, we adopt bi-affine in the the following experiments, computed as:

$$\alpha = \text{Sigmoid}(h_{name}^{T}(s) W_{\text{biaf}} h_{ctx}(s)) \quad (4)$$

Finally, $h(s)$ is fed into a feed forward layer followed by the Softmax function to get the probability distribution $p(x)$. Note that if there are two spans with overlapping parts, we only leave the one with higher $max(p(x))$. We use cross-entropy as loss function during training:

$$\mathcal{L}_{\text{com}} = \sum_{s\in\mathcal{S}} \sum_{l\in L} \left(-y(s)^{(l)} \log(p(s)^{(l)})\right) \quad (5)$$

where $\mathcal{S}$ is the set of all candidate spans in the sentence.

If we are able to give α a gold label, we can also compute a negative log-likelihood as the loss $\mathcal{L}_{\alpha}$ for α. Then, for each span candidate in the sentence, we can derive a loss function that is the sum of both parts: $\mathcal{L} = \mathcal{L}_{\alpha} + \mathcal{L}_{\text{com}}$. However, we do not have labels for α in current CNER datasets. The supervision on α to explicitly guide the model when to focus on features from the context is significant. We will introduce it in the next section.

3.4 Tailored Data Augmentations

We propose the following data augmentation strategies that assign different labels for name and context modules on the same sample. The augmentations strategies aim to guide the model focus on the name or context information correctly. Some strategies provide a label for α, turn the learning on α into *weak-supervised* from *un-supervised*.

As summarized in Table 1, the augmentation strategies are categorized into two groups: insertion and replacement. We assign labels for different strategies: For each module, ✓ means we keep the original label (a positive entity type), ✗ means we set it as negative (None type), "-" means ignore the loss for this module in current sample.

The first class of augmentation is replacement. Suppose we are given a sentence x and a span $s = (w_i, ..., w_j)$, $l(s) \in E$, i.e. s is an entity mention. We can replace s by another token sequence $s' = (w'_1, ..., w'_m)$ to get a new sentence:

$$x' = (w_1, ..., w_{i-1}, w'_1, ..., w'_m, w_{j+1}, ..., w_n).$$

For the Mention Replacement strategy, s' is an entity mention with the same type as s. For the Random Replacement strategy, s' is a random string. So, we set the label for name module as None, but set the label for context module as $l(s)$. Dai and Adel (2020)'s experiments show that attempting to improve the NER performance by using only a single mention of substitution does not yield significant improvements. We believe that the replacement for augmentation is not enough.

Another class of augmentation strategies is insertion. We insert a text span s' into the sentence at random position. For the Random Insertion strategy, s' is not an entity mention, the model should predict it as negative (i.e. $\hat{y}(s') =$ None). For the Mention Insertion strategy, s' is an entity mention in another sentence. We set label for name module as $\hat{y}(s)$, and label for context as None, to encourage the name module focus on the name.

As for α, we set its label lean to context in "Replacement (Random)" to encourage the model to predict unseen mentions[2]. We set its label to lean to name to accept clear name patterns with unclear contexts. In summary, we set different labels to guide the model focus on targeted information. One possible concern about these strategies is that the random replacement and random insertion strategies generate confusing and unreasonable samples. Such samples are difficult to be incorporated by most existing models, as we cannot assign them a reasonable type. However, we take them in our model to guide the focus of modules, and to simulate the extreme cases of rare context or mentions. It means the unreasonable augmented samples are only treated as extra annotations for training specific modules. We do not utilize these counter-intuitive samples in evaluation.

4 Invariance Test

The performance of the NER model is overestimated with traditional metrics on a fixed test set. We propose an evaluation method and the corresponding metric which is an extension of "Invariance Test" from the "Checklist" (Ribeiro et al.,

[2]Each character in the random string does not exceed the character set of the current data set.

Strategy	Converted Sample	Label			
		name	ctx	com	α
*For a sentence with entity mention span: Aunt **Kate** ate this cake.*					
Boundary Movement	Aunt **Kate ate** this cake.	✗	✗	✗	-
Mention Replacement	Aunt **John Smith** ate this cake.	✓	✓	✓	-
Mention Replacement (Random)	Aunt **Vodka** ate this cake.	✗	✓	-	ctx
For a sentence not specifying the span: The cake tastes great.					
Mention Insertion	The **John Smith** cake tastes great.	✓	✗	-	name
Mention Insertion (Random)	The cake tastes **Vodka** great.	✗	✗	-	-

Table 1: Augmentation strategies. The bold blue tokens are span candidates. ✓ and ✗ are labels for modules, "-" means ignoring the loss. For each kind of module, we ensure that both positive and negative samples exist.

Dataset	MSRA	Resume	Onto4
# Train Samples	45000	3821	15724
# Valid Samples	-	463	4301
# Test Samples	3442	477	4346
# Tag Types	3	8	4
Avg # Chars	48.22	32.48	31.28
Avg # Entities	1.67	3.52	0.85
Avg Ent Length	3.23	5.88	3.08

Table 2: Statistics of three datasets. All Average measures are calculated on training samples.

2020). It is similar to Mention Replacement data augmentation introduced in Section 3.4. As we replace a mention with another mention with the same type, the resulting samples usually have correct grammar. At first, We randomly select n_{it} sentences from the test set. For each entity type $t \in \mathcal{E}_{it}$, we collect all mentions M in these sentence. Then, for the X sentences that contain a mention with type t, we replace the first one with all mentions in M. The process generates $|M| \times |X|$ manipulated mentions to be recognized. The NER models are expected to recognize and classify as type t correctly.

5 Experiments

5.1 Datasets

In this section, we will introduce the widely-used (Robert et al., 2021b) CNER datasets. Details for these datasets are shown in Table 2.

• **MSRA dataset**. MSRA (MicroSoft Research Asia (2006)) is the simplified Chinese version of the Microsoft NER dataset on the Third SIGHAN Chinese Language Processing Bakeoff [3]. Original MSRA dataset only has training set and testing set,

[3]http://sighan.cs.uchicago.edu/bakeoff2006/

• **Resume dataset**. The Resume dataset is another popular Chinese NER dataset proposed by Zhang and Yang (2018).

• **Onto4 dataset**. The OntoNote 4 dataset is also a well-known Chinese NER dataset released by Weischedel et al. (2011).

5.2 Baselines

We compare our model with the following models.

• **LSTM-CRF** and **BERT-CRF.** We take Bi-LSTM CRF (Lample et al., 2016) and BERT CRF (Devlin et al., 2018) as baseline models which is the de facto standard sequence labeling model for NER. Moreover, we utilize them as the encoders for two modules in our Model, without CRF parts.

• **LatticeLSTM** and **FLAT.** One of the best solutions for sentence-level NER in Chinese is with the lattice structure (Zhang and Yang, 2018) as lexical information. FLAT (Li et al., 2020) achieves the state-of-the-art result on MSRA dataset.

• **PURE-CN.** As the best performing span-based solution in English NER, PURE (Zhong and Chen, 2021) outperforms on English NER datasets like ACE and GENIA. We follow the span-based method from this work and adapt it for CNER.

• **Others.** In Chinese NER, using the glyph information as an additional kind of embedding of characters is also effective. Such as **Glyce** (Meng et al., 2019) and FGN (Xuan et al., 2020). This kind of solution, together with the aforementioned **FLAT**, are the current best solutions (Robert et al., 2021a) on the CNER task. However, the pre-trained glyph embeddings are not released yet, so we cannot reproduce their results and conduct invariance tests on them.

5.3 Experiment Settings

Embeddings, hyper-parameters and metrics are carefully designed for a fair comparison.

• **Embeddings.** The parameters for embeddings in LSTM are initialized from the same BERT model[4]. It means the token embeddings in different modules or models are calculated with the same position embeddings, layer normalization and so on. In NC Model, the embeddings in BERT are freezed during the training.

• **Hyper-Parameters.** For all Chinese datasets, we set batch size as 4, L_{sp} as 25, and conduct 10 epochs of training. The learning rate for modules in our model is 5e-4, wheile BERT's learning rate is 1e-5. Moreover, we set the dropout rate to 0.2 for FFN. The model takes a step of 50 to find $\theta_{\text{top-sp}}$, and $\theta_{\text{top-sp}} = 300$ is the largest before Out-Of-Memory problem occurs[5]. The selected spans cover more than 98% entiies for all CNER datasets we mentioned in Section 5.1.

• **Evaluation Metrics.** We adopt micro F1-Score as the evaluation metric. Only when both the span (l, r) and the entity type t are correct at the same time, we count it as a hit. Following the definition from Xiao et al. (2019), we treat any spans that do not exactly match the annotation as negative (it affects precision). It means that entities which are not recalled from the span generation step are counted as misses. Nested mentions, such as the "Nanjing" in the organization mention "Nanjing University" is regarded as negative, as it does not appear in the annotations.

Another evaluation metric is "Failure Rate" for the invariance tests mentioned in Section 4. We propose this new metric to measure a model :

$$\mathbf{FR} = \frac{1}{|S|} \sum_{s_i \in S} \mathbb{1}(\hat{y}(s_i) \neq y(s_i)) \quad (6)$$

here $\mathbb{1}(\cdot)$ is the indicator function.

5.4 Overall Results

The experimental results on different dataset are shown in Table 3. As shown in Table 3, current state of the art models heavily depend on external features: 1) the in-house corpus and glyph features[6] are not publicly available yet[7], making it difficult to reproduce or improve upon the **Glyce** model. 2) **FLAT** gathers over 700 thousand external pre-trained word embeddings from 3 different external resources for lexicon features. The model performance heavily relies on the quality of lexicons and plenty of n-gram embeddings.

Our model utilizes BERT only, and it is exciting that our model performs comparably to these best CNER solutions. The ability to rival the optimal CNER method without the need for external features and tools makes our model easy to use in practical applications. Compared to BERT-CRF with the same inputs at prediction, our model improves 1.2% on average.

To figure out why the model improves performance, we put attention on ablations and the invariance test in the following sections.

Dataset	Model	Performance P	R	F1
MSRA	BiLSTM-CRF (2016)	93.0	90.8	91.9
	LatticeLSTM$^\diamondsuit$ (2018)	94.0	92.2	93.1
	BERT-CRF (2018)	95.0	93.3	94.1
	PURE-CN (2021)	95.6	95.3	95.4
	NC (Ours)	95.7	95.0	95.4
	Glyce$^\spadesuit$ (2019)	95.6	95.5	95.5
	FLAT$^\diamondsuit$ (2020)	-	-	**96.1**
Resume	BiLSTM-CRF	94.5	94.3	94.4
	LatticeLSTM$^\diamondsuit$	94.8	94.1	94.5
	BERT-CRF	94.0	95.2	94.6
	PURE-CN	95.5	96.0	95.8
	NC (Ours)	96.2	96.5	96.3
	Glyce$^\spadesuit$	96.6	96.5	**96.5**
	FLAT$^\diamondsuit$	-	-	95.9
Onto4	BiLSTM-CRF	74.4	69.4	71.8
	LatticeLSTM$^\diamondsuit$	74.2	73.1	73.6
	BERT-CRF	82.4	78.4	80.4
	PURE-CN	83.5	78.5	80.9
	NC (Ours)	80.9	81.1	81.0
	Glyce$^\spadesuit$	81.9	81.4	81.6
	FLAT$^\diamondsuit$	-	-	**81.8**

Table 3: Overall Results on different datasets. The network of BERT in "BERT-CRF" is optimized by us so that it performs better than original. We take the same BERT in our approaches. $\diamondsuit$ for utilizing external lexicon and embeddings and $\spadesuit$ for glyph embeddings.

5.5 Ablations on Augmentation Strategies

We take the ablation experiment on CNER datasets for further analysis as shown in Table 4. In this section, we test the validity of our proposed augmentation strategies.

In Table 4, we take ablation tests on augmentation strategies. Moreover, we also calculate FR to figure out their contribution for the NER model's generalization ability. These CNER datasets share

[4]"bert-chinese_L-12_H-768_A-12", while FLAT uses bert-wwm with the same embedding size as its default setting.

[5]on single Nvidia 12G RTX 2080Ti

[6]pre-trained Glyph embeddings and Glyce-BERT

[7]github.com/ShannonAI/glyce/issues

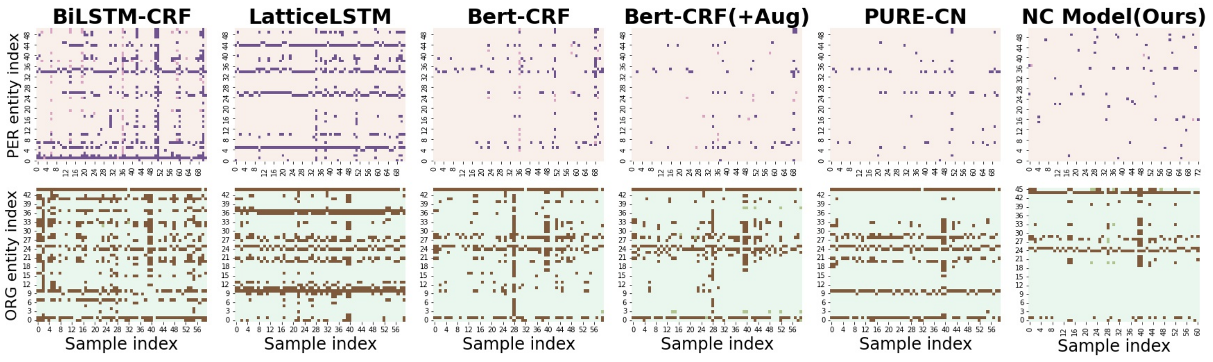

Figure 3: Invariance test by combining different entity mentions and contexts. In the subfigures, the columns are for different models and the rows are for different entity types (**PER**son and **ORG**anization).

Model	MSRA F1	MSRA FR	Resume F1	Resume FR	Onto4 F1	Onto4 FR
NC Model	**95.4**	4.83	**96.3**	**2.73**	**81.0**	**21.79**
- Aug	95.0	**4.53**	96.0	2.79	80.6	22.53
- Replacement	95.0	5.14	96.2	2.85	79.9	22.54
- Insertion	95.2	4.74	95.8	3.39	80.5	24.08

Table 4: Ablations on augmentation strategies for NC Model on CNER datasets. "-" means taking the part away.

Models	BiLSTM -CRF	Latt -ice	BERT-CRF Origin	BERT-CRF (+Aug)	PURE -CN	NC (Ours)
PER	14.5	6.7	3.4	1.9	2.3	**1.6**
ORG	25.1	23.4	14.5	14.6	14.3	**9.3**
Total	19.0	13.6	8.1	7.3	7.4	**4.8**

Table 5: Failure rate (**FR**) in Invariance Test on the MSRA dataset. We also tried to apply data augmentations (+Aug) on BERT-CRF.

a subset of tags {PER, ORG}, we treat it as $\mathcal{E}_{it}$.

When removing these augmentation strategies, both F1-Score and FR decrease. The model learn when to depend on context information without labels. Two kinds of strategies generate different labels for α, the absence of either "name" label or "context" label makes it difficult for α to function. On one hand, if there is no "context" labels, the model is not aware of when it needs to rely on the context module. On the other hand, the "name" labels teach the model when to discard all the features from the context module. In Section 3.4, we argue that the augmentation needs the insertion strategy in addition to the replacement strategy. The experimental results proved it is reasonable.

5.6 Invariance Test

We show the results of invariance tests as a heatmap in Figure 3. We take the MSRA dataset as an example to show the corresponding statistics in Table 5.

In the heatmap, all dots in each sub-figure indicate model prediction failures, and faded dots indicate a faulty tag type for a correctly recalled span. Note that the not-recalled spans in our model are also counted as failures. A large number of dots in a row means the entity name is hard for the model, while many dots in a column means the context is confusing for the model. On the other hand, fewer dots indicates that the corresponding model generalizes better and produces consistent results under manipulation above. We also try to apply data augmentations for sequence labeling models. For instance, the failure rate of BERT-CRF drops 0.8% (relatively reduced by 9.9%) with our proposed data augmentations. It indicates that data augmentation also works on sequence labeling models. Our proposed model performs much better than other models with a failure rate of 4.8%. We can see that the invariance performance of the model can be ranked as:

NC Model > BERT-CRF (+Aug) ≈ PURE-CN ≈ BERT-CRF > LatticeLSTM > BiLSTM-CRF.

6 Conclusions and Future Work

We propose a model containing both the name and context channels of an entity mention. A balance factor for the channels guides whether the model depends on the name or context for classification. To help the model learn the factor and modules, we also propose data augmentation strategies. As a result, our model performs on par with the best-performing ones on the CNER task and outperforms on the invariance test.

Further improvements can be expected if each module can be replaced with more efficient ones. We leave this to future work.

References

Oshin Agarwal, Byron C. Wallace, Yinfei Yang, and Ani Nenkova. 2020a. Entity-switched datasets: An approach to auditing the in-domain robustness of named entity recognition models. *ArXiv*.

Oshin Agarwal, Yinfei Yang, Byron C. Wallace, and Ani Nenkova. 2020b. Interpretability Analysis for Named Entity Recognition to Understand System Predictions and How They Can Improve. *ArXiV*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *ArXiv*.

Yixuan Cao, Dian Chen, Hongwei Li, and Ping Luo. 2019. Nested Relation Extraction with Iterative Neural Network. In *CIKM*, pages 1001–1010, New York, New York, USA. ACM Press.

Xiang Dai and Heike Adel. 2020. An Analysis of Simple Data Augmentation for Named Entity Recognition. In *COLING*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv*.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-Based LSTM-CRF with Radical-Level Features for Chinese Named Entity Recognition. In *Lecture Notes in Computer Science*, volume 10102, pages 239–250.

Markus Eberts and Adrian Ulges. 2019. Span-based Joint Entity and Relation Extraction with Transformer Pre-training. *Frontiers in Artificial Intelligence and Applications*, 325:2006–2013.

Anjalie Field and Yulia Tsvetkov. 2019. Entity-Centric Contextual Affective Analysis. In *ACL*, pages 1462–1467, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jinlan Fu, Pengfei Liu, and Qi Zhang. 2020. Rethinking Generalization of Neural Models: A Named Entity Recognition Case Study. *AAAI*, 34(05):7732–7739.

Tao Gui, Xiao Wang, Qi Zhang, Qin Liu, Yicheng Zou, Xin Zhou, Rui Zheng, Chong Zhang, Qinzhuo Wu, Jiacheng Ye, Zexiong Pang, Yongxin Zhang, Zhengyan Li, Ruotian Ma, Zichu Fei, Ruijian Cai, Jun Zhao, Xinwu Hu, Zhiheng Yan, Yiding Tan, Yuan Hu, Qiyuan Bian, Zhihua Liu, Bolin Zhu, Shan Qin, Xiaoyu Xing, Jinlan Fu, Yue Zhang, Minlong Peng, Xiaoqing Zheng, Yaqian Zhou, Zhongyu Wei, Xipeng Qiu, and Xuanjing Huang. 2021. TextFlint: Unified Multilingual Robustness Evaluation Toolkit for Natural Language Processing. *ArXiV*, pages 1–28.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *ArXiV*.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Span-BERT: Improving Pre-training by Representing and Predicting Spans. *TACL*, 8:64–77.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *NAACL-HLT*, pages 260–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *EMNLP*, pages 188–197, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. FLAT: Chinese NER Using Flat-Lattice Transformer. In *ACL*, volume 1, pages 6836–6842, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. BOND: BERT-Assisted Open-Domain Named Entity Recognition with Distant Supervision. In *KDD*, pages 1–23.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020a. TriggerNER: Learning with Entity Triggers as Explanations for Named Entity Recognition. In *ACL*, pages 8503–8511. Association for Computational Linguistics.

Hongyu Lin, Yaojie Lu, Jialong Tang, Xianpei Han, Le Sun, Zhicheng Wei, and Nicholas Jing Yuan. 2020b. A Rigorous Study on Named Entity Recognition: Can Fine-tuning Pretrained Model Lead to the Promised Land? In *EMNLP*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *ACL*, volume 2, pages 1064–1074, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yuxian Meng, Wei Wu, Fei Wang, Xiaoya Li, Ping Nie, Fan Yin, Muyu Li, Qinghong Han, Xiaofei Sun, and Jiwei Li. 2019. Glyce: Glyph-vectors for Chinese Character Representations. In *NeurIPS*, pages 1–12.

Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2020. Learning from Context or Names? An Empirical Study on Neural Relation Extraction. In *EMNLP*, pages 3661–3672, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In *ACL*, pages 4902–4912, Stroudsburg, PA, USA. Association for Computational Linguistics.

Robert, Ross, Marcin, Viktor, Ludovic, Elvis, and Guillem. 2021a. Chinese Named Entity Recognition on MSRA.

Robert, Ross, Marcin, Viktor, Ludovic, Elvis, and Guillem. 2021b. Chinese Named Entity Recognition Tasks.

Karl Stratos. 2017. Entity Identification as Multitasking. In *Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing*, pages 7–11, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary Enhanced Neural Span Classification for Nested Named Entity Recognition. In *AAAI*, volume 34, pages 9016–9023.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, Relation, and Event Extraction with Contextualized Span Representations. In *EMNLP-IJCNLP*, pages 5783–5788, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2011. OntoNotes Release 4.0 LDC2011T03.

Congying Xia, Chenwei Zhang, Tao Yang, Yaliang Li, Nan Du, Xian Wu, Wei Fan, Fenglong Ma, and Philip Yu. 2020. Multi-grained named entity recognition. In *ACL*, pages 1430–1440.

Shiyuan Xiao, Yuanxin Ouyang, Wenge Rong, Jianxin Yang, and Zhang Xiong. 2019. Similarity Based Auxiliary Classifier for Named Entity Recognition. In *EMNLP*, pages 1140–1149, Hong Kong, China.

Ji Xin, Hao Zhu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2018. Put It Back: Entity Typing with Language Model Enhancement. In *EMNLP*, pages 993–998, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mingbin Xu, Hui Jiang, and Sedtawut Watcharawittayakul. 2017. A Local Detection Approach for Named Entity Recognition and Mention Detection. In *ACL*, volume 1, pages 1237–1247, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhenyu Xuan, Rui Bao, and Shengyi Jiang. 2020. FGN: Fusion Glyph Network for Chinese Named Entity Recognition. *ArXiv*.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2015. Corpus-level Fine-grained Entity Typing Using Contextual Information. In *EMNLP*, pages 715–725, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yue Zhang and Jie Yang. 2018. Chinese NER Using Lattice LSTM. In *ACL*, volume 1, pages 1554–1564, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Yuying Zhu, Guoxin Wang, and Börje F. Karlsson. 2019. CAN-NER: Convolutional Attention Network for Chinese Named Entity Recognition. In *NAACL-HLT*, volume 1, pages 3384–3393.