# E-NER: Evidential Deep Learning for Trustworthy Named Entity Recognition

**Zhen Zhang**[1] **Mengting Hu**[1*] **Shiwan Zhao**[†] **Minlie Huang**[2] **Haotian Wang**[1]
**Lemao Liu**[3] **Zhirui Zhang**[3] **Zhe Liu**[4] **Bingzhe Wu**[3*]
[1] College of Software, Nankai University, [2] The CoAI group, Tsinghua University
[3] Tencent AI Lab, [4] Zhejiang Lab
zhangz@mail.nankai.edu.cn, mthu@nankai.edu.cn

## Abstract

Most named entity recognition (NER) systems focus on improving model performance, ignoring the need to quantify model uncertainty, which is critical to the reliability of NER systems in open environments. Evidential deep learning (EDL) has recently been proposed as a promising solution to explicitly model predictive uncertainty for classification tasks. However, directly applying EDL to NER applications faces two challenges, i.e., the problems of *sparse entities* and *OOV/OOD entities* in NER tasks. To address these challenges, we propose a trustworthy NER framework named E-NER [1] by introducing two uncertainty-guided loss terms to the conventional EDL, along with a series of uncertainty-guided training strategies. Experiments show that E-NER can be applied to multiple NER paradigms to obtain accurate uncertainty estimation. Furthermore, compared to state-of-the-art baselines, the proposed method achieves a better OOV/OOD detection performance and better generalization ability on OOV entities.

## 1 Introduction

Named entity recognition (NER) aims to locate and classify entities in unstructured text, such as extracting LOCATION information *"New York"* from the sentence *"How far is New York from me"*. Thanks to the development of deep neural network (DNN), current NER methods have achieved remarkable performance on a wide range of benchmarks (Lample et al., 2016; Yamada et al., 2020; Li et al., 2022).

Despite this progress, current NER-related research typically focuses on improving the model performance, such as recognition accuracy and F1 scores (Yu et al., 2020; Zhu and Li, 2022).

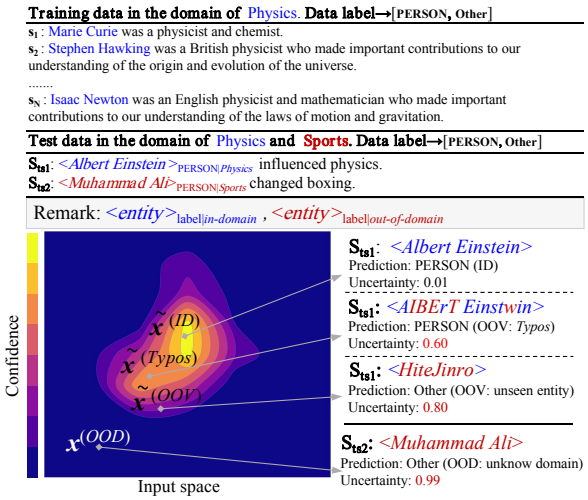



Figure 1: Visualization of desired uncertainty estimations in the NER application.

However, seldom works focus on investigating the model's reliability. The critical aspect of the model reliability is the uncertainty estimation of the predictive results, which can characterize the probability that the model prediction will be wrong. One natural way to construct the predictive uncertainty is based on the maximum value of the Softmax output (Yan et al., 2021; Li et al., 2022; Zhu and Li, 2022) (the smaller this value, the larger the uncertainty). However, previous empirical studies show that probabilistic predictions produced by DNN models (e.g., transformer and CNN) are often inaccurate (Guo et al., 2017; Lee et al., 2018; Pinto et al., 2022). Therefore, this natural way may over/under-estimate the predictive uncertainty, hindering the model's reliability.

High-quality uncertainty estimation helps to improve the model's reliability in an open environment and to find valuable samples to improve training sample efficiency, thus reducing the cost of manual labeling. On the one hand, for the reliability aspect, accurate uncertainty estimation can equip the NER model with the ability to express

* Mengting Hu and Bingzhe Wu are the corresponding authors.
† Independent researcher.
[1] https://github.com/Leon-bit-9527/ENER



*Findings of the Association for Computational Linguistics: ACL 2023*, pages 1619–1634
July 9-14, 2023 

*"I do not know"* to both the out-of-domain (OOD) or out-of-vocabulary (OOV) samples (Charpentier et al., 2020). A desired uncertainty estimation is conceptually shown in Figure 1, wherein misclassified OOV/OOD entities are assigned with significantly higher uncertainty than the in-domain (ID) entities. Besides, the estimated uncertainty can be further absorbed into the training process to improve the model robustness against OOV/OOD samples. On the other hand, for the sample efficiency aspect, prior work shows that high-quality uncertainty estimation can also be used for selecting more "informative" samples and thus can reduce the number of labeled samples required for training the NER model.

To attain high-quality uncertainty estimation, evidential deep learning (EDL) (Sensoy et al., 2018) provides a promising solution. EDL is superior to existing Bayesian learning-based methods (Blundell et al., 2015; Kingma et al., 2015; Graves, 2011) in that model uncertainty can be efficiently estimated in a single forward pass that avoids inexact posterior approximation (Kopetzki et al., 2021) or time/storage-consuming Monte Carlo sampling (Gal and Ghahramani, 2016). However, directly applying conventional EDL to NER applications still faces two critical challenges: (1) *sparse entities*: In text corpus, entities only take a minority. For example, only 16.8% of the words in the commonly used CoNLL2003 dataset belong to entities. The remaining non-entity types are labeled into the "others" (O) class. The imbalance between entity and non-entity words can cause over-fitting and poor performance on the entity types. (2) *OOV/OOD entity discrimination*: In the open environment, NER training/test data typically comes with OOV/OOD entities. However, the optimization objective of current EDL methods lacks explicit modeling of such types of information.

To address these two issues, we present a trustworthy NER framework named E-NER with a series of uncertainty-guided training strategies. For the issue of sparse entities, we propose to use an uncertainty-guided importance weighted (IW) loss, wherein samples with higher predictive uncertainties are assigned larger weights. This loss helps the model training to pay more attention to entities of interest (e.g., person and location). To solve the issue of unknown entities, we present an additional regularization term to penalize the case where labels are more prone to errors by assigning higher uncertainties to corresponding samples. We empirically show these two uncertainty-guided loss terms can improve both the quality of estimated confidence and the robustness against OOV samples.

Our contributions are summarized as follows:

- To the best of our knowledge, E-NER is the first work to explore how to leverage evidential deep learning to improve the reliability of current NER models. This work has successfully shown the potential of EDL to provide high-quality uncertainty estimation in NER applications. The estimated uncertainty can be further used for detecting OOD/OOV samples in the test phase.
- For the technique contribution, we propose two uncertainty-guided loss terms to mitigate sparse entities and OOV/OOD entity discrimination issues in the NER task.
- E-NER is extensively validated in a series of experiments. In contrast to conventional NER methods, the result shows that E-NER comes with the following superiority: (1) more accurate uncertainty estimation. (2) better OOV/OOD detection performance. (3) better generalization ability on OOV entities. (4) better sample efficiency (i.e., fewer samples are required to achieve the same-level performance).

## 2 Preliminary

This section introduces a commonly-used EDL implementation based on the Dirichlet-based model (DBM) (Sensoy et al., 2018). We then describe how the DBM computes the uncertainty in a closed form.

### 2.1 Dirichlet-based Model

Conventional neural network classifiers typically employ a Softmax layer to provide a point estimation of the categorical distribution. In contrast, Dirichlet-based models (DBM) output the parameters of a Dirichlet distribution and then use it to estimate the categorical distribution. Specifically, for the $i$-th sample $x^{(i)}$ (e.g., the $i$-th word in the NER task) in the $C$-class classification task, the DBM replaces the Softmax of the neural network with an activation function layer (e.g., Softplus) to ensure that the network outputs non-negative values, which are considered as the evidence $\mathbf{e}^{(i)} \in \mathbb{R}_+^C$

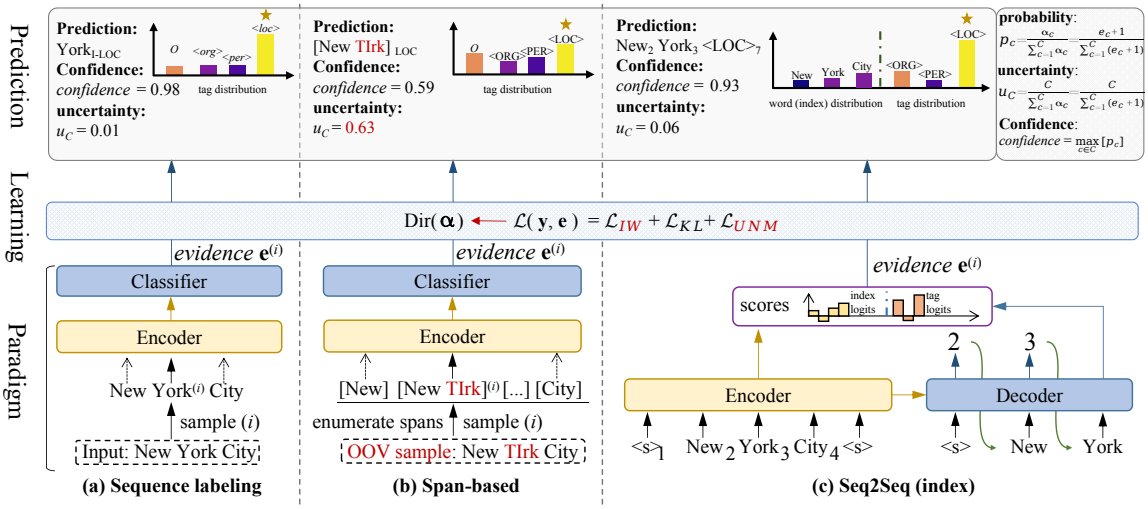



Figure 2: Basic architecture of E-NER with different NER paradigms.

to support the classification. The evidence is then used for constructing a Dirichlet distribution which models the distribution over different classes. To this end, the parameter of a Dirichlet distribution is obtained by: $\boldsymbol{\alpha}^{(i)} = \mathbf{e}^{(i)} + \mathbf{1}$, where $\mathbf{1}$ represents the vector of $C$ ones. Finally, the density function of Dirichlet distribution is given by:

$$\mathrm{Dir}(\mathbf{p}^{(i)}|\boldsymbol{\alpha}^{(i)}) = \frac{1}{B(\boldsymbol{\alpha}^{(i)})} \prod_{c=1}^{C} p_c^{(\alpha_c^{(i)}-1)}, \quad (1)$$

where $B(\boldsymbol{\alpha}^{(i)})$ is the $C$-dimensional multinomial beta function.

To learn model parameters, given the sample $(x^{(i)}, \mathbf{y}^{(i)})$, where $\mathbf{y}^{(i)}$ is a one-hot $C$-dimensional label for sample $x^{(i)}$, previous EDL methods build the optimization objective by combining a cross-entropy classification loss $\mathcal{L}_{CLS}$ and a KL penalty loss $\mathcal{L}_{KL}$:

$$\begin{aligned}\mathcal{L}_{EDL}^{(i)} &= \mathcal{L}_{CLS}^{(i)} + \mathcal{L}_{KL}^{(i)} \\ &= \underbrace{\sum_{c=1}^{C} y_c^{(i)} \left(\psi(S^{(i)}) - \psi(\alpha_c^{(i)})\right)}_{\text{(a) classification loss}} \\ &+ \underbrace{\lambda_1 KL[\mathrm{Dir}(\mathbf{p}^{(i)}|\widetilde{\boldsymbol{\alpha}}^{(i)})||\mathrm{Dir}(\mathbf{p}^{(i)}|\mathbf{1})]}_{\text{(b) penalty loss}},\end{aligned} \quad (2)$$

where $\psi(\cdot)$ is the digamma function, and $S^{(i)} = \sum_{c=1}^{C} \alpha_c^{(i)}$ denotes the Dirichlet strength, $\lambda_1$ is the balance factor, $\mathrm{Dir}(\mathbf{p}^{(i)}|\mathbf{1})$ is a special case which is equivalent to the uniform distribution, and $\widetilde{\boldsymbol{\alpha}}^{(i)} = \mathbf{y}^{(i)} + (1 - \mathbf{y}^{(i)}) \odot \boldsymbol{\alpha}^{(i)}$ denotes the masked parameters while $\odot$ refers to the Hadamard (element-wise) product, which removes the non-misleading evidence from predicted parameters $\boldsymbol{\alpha}^{(i)}$. Intuitively, the first term in Eq. 2 measures the classification performance while the second term can be seen as a regularization term that penalizes misleading evidences by encouraging the associate distribution to be close to uniform distribution (see more details in Appendix §C.3).

### 2.2 Uncertainty Estimation of DBM

Once we obtain the Dirichlet distribution for prediction, we can estimate the predictive uncertainty in a closed form. To this end, EDL provides two probabilities: *belief mass* and *uncertainty mass*. The belief mass $\mathbf{b}$ represents the probability of evidence assigned to each category and the uncertainty mass $u$ provides uncertainty estimation. Specifically, for the sample $x^{(i)}$, the belief mass $b_c^{(i)}$ and uncertainty $u^{(i)}$ are computed as:

$$b_c^{(i)} = \frac{e_c^{(i)}}{S^{(i)}} \quad \text{and} \quad u^{(i)} = \frac{C}{S^{(i)}}, \quad (3)$$

with the restrictions that $u^{(i)} + \sum_{c=1}^{C} b_c^{(i)} = 1$. The belief mass $\mathbf{b}$ and the uncertainty mass $u$ will be used to guide the training process in our proposed framework (see Section §3.3).

## 3 E-NER Architecture

In this section, we describe the three core modules of E-NER and provide an overview of the system architecture in Figure 2. Additionally, we revise the learning strategy of EDL by incorporating importance weights (IW) to address the sparse entities

problem and uncertainty mass optimization (UNM) to model the uncertainty of mispredicted entities.

### 3.1 NER Feature Extraction

Given a word sequence $X = \{x^{(1)}, ..., x^{(n)}\}$ and a target sequence $Y = \{y^{(1)}, ..., y^{(n)}\}$. To obtain the hidden representation $H$ of $X$, the words in the sentence $X$ are first preprocessed according to the input form required by the corresponding NER method. Then the processed input is fed into an Encoder module (e.g., BERT (Devlin et al., 2019)) to compute the hidden representation $H = \text{Encoder}(X)$, where $H \in \mathbb{R}^{n \times d_h}$ and $d_h$ denotes the dimension of the hidden representation. The input format for NER models can vary depending on the paradigm used. Three NER paradigms were considered for this study: sequence labeling (Figure 2(a)), span-based (Figure 2(b)), and Seq2Seq (Figure 2(c)). The specific formats for these paradigms are provided in the Appendix §A. Note that in the Seq2Seq (sequence-to-sequence) paradigm, we choose a pointer-based model (Yan et al., 2021), so that we don't need to learn on the entire vocabulary.

### 3.2 Dirichlet-based Prediction Layer

Once we obtain the hidden representation, we introduce a Dirichlet-based layer to produce the final predictive distribution. Precisely, for the $i^{th}$ sample, the hidden representation $h$ is fed to the fully connected layer to output logits, and then we can transform the logits into Dirichlet parameters $\boldsymbol{\alpha}$ as described in Section §2.1. Finally, as shown in Figure 2, only one forward step using Eq. 3 is sufficient to calculate the uncertainty $u^{(i)}$, while the probability distribution $\mathbf{p}^{(i)}$ and prediction $y^{(i)}$ are calculated as follows:

$$\mathbf{p}^{(i)} = \frac{\boldsymbol{\alpha}^{(i)}}{S^{(i)}}, \qquad y^{(i)} = \arg\max_{c \in C} \left[ p_c^{(i)} \right]. \tag{4}$$

### 3.3 E-NER Model Learning

**Overview**. The objective function of EDL training is to minimize the sum of losses over all words. Due to the *sparse entities* and *OOV/OOD entities* issues, directly applying EDL to NER leads suboptimal uncertainty estimates. We improve conventional EDL methods by incorporating belief mass and uncertainty into the network training process. Specifically, two key modifications are introduced: (1) We compute importance weights for each sample based on the belief mass to reweight the original classification loss in Eq. 2(a). (2) We introduce an additional term to increase the uncertainty of mispredicted instances, which explicitly improves the quality of uncertainty estimation and helps OOD entity detection.

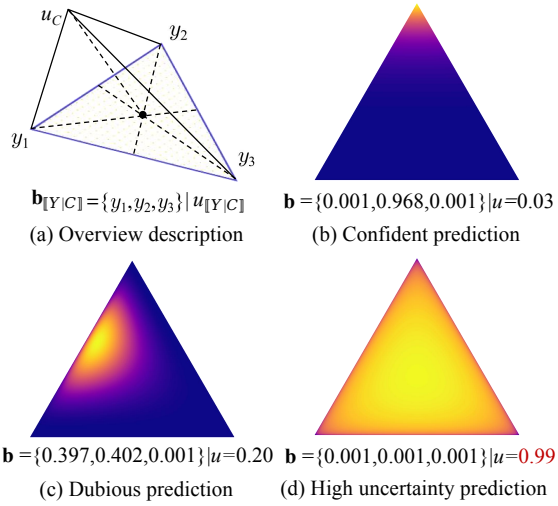



Figure 3: (a) Overview of uncertainty estimation for Dirichlet distributions. (b-d) Typical patterns of Dirichlet distribution for an example 3-class classification task.

**Importance Weight**. Due to the inherent imbalance between entities and non-entities in NER datasets, conventional EDL methods tend to overfit non-entities and assign high uncertainty estimates to entities. To make the training focus more on the entities and increase the evidence corresponding to the ground-truth category, we use the belief mass of the ground-truth category to compute the category-level uncertainty for each instance to adjust the loss. Specifically, for the $i^{th}$ sample, we use $(\mathbf{1} - \mathbf{b}^{(i)})$ as the category-level uncertainty which serves as the importance weights of entity categories during training. To this end, we replace the ground truth $\mathbf{y}^{(i)}$ of one-hot representation with an importance weight (IW) $\mathbf{w}^{(i)} = (\mathbf{1} - \mathbf{b}^{(i)}) \odot \mathbf{y}^{(i)}$, and lastly, the Eq. 2(a) is adjusted to:

$$\mathcal{L}_{IW}^{(i)} = \sum_{c=1}^{C} w_c^{(i)} \left( \psi(S^{(i)}) - \psi(\alpha_c^{(i)}) \right). \tag{5}$$

As illustrated in Figure 3(b), the belief mass of the ground-truth category is high, indicating a high level of certainty in the prediction. In this case, the importance weight (IW) assigned will be small. Conversely, Figure 3(c) presents a small belief mass, indicating an uncertain prediction. IW will be assigned a large value. In this manner, the

training process can focus more on sparse but valuable entities.

**Uncertainty Mass Optimization**. Assigning high uncertainty to OOV/OOD entities (see Figure 3(d) as an example) facilitates OOV/OOD entity detection. However, ground-truth OOV/OOD samples are not available during training. One solution is to synthesize such data on the boundary of the in-domain region via a generative model (Lee et al., 2018). In this paper, we propose a more convenient way to treat hard samples as OOV/OOD samples which are often outliers and are mispredicted even after adequate model training. In this way, we enable the model to detect OOV/OOD data. Specifically, uncertainty mass optimization (UNM) assigns higher uncertainty to more error-prone samples for the model to express a lack of evidence, by adding an uncertainty mass penalty term $\mathcal{L}_{UNM}$ to the wrongly predicted samples:

$$\mathcal{L}_{UNM} = -\lambda_2 \sum_{i \in \{\hat{y}^{(i)} \neq y^{(i)}\}} \log(u^{(i)}). \quad (6)$$

The coefficient $\lambda_2 = \lambda_0 \exp\{-(\ln\lambda_0/T)t\}$, where $\lambda_2 \in [\lambda_0, 1]$, $\lambda_0 \ll 1$ is a small positive constant, $t$ is the current training epoch, and $T$ is the total number of training epochs. As the training epoch $t$ increases towards $T$, the factor $\lambda_2$ will increase monotonically from $\lambda_0$ to 1.0. This allows the network to initially focus on optimizing classification and gradually shift its emphasis towards optimizing UNM as the training progresses.

**Overall Loss**. The overall loss function combines three components: the importance weighted classification loss $\mathcal{L}_{IW}$, the KL divergence penalty loss $\mathcal{L}_{KL}$, and the uncertainty mass loss $\mathcal{L}_{UNM}$ for mispredicted entities. Each element contributes to the overall loss and is defined as follows:

$$\mathcal{L}_{overall} = \sum_{i=1}^{N} (\mathcal{L}_{IW}^{(i)} + \mathcal{L}_{KL}^{(i)}) + \mathcal{L}_{UNM}. \quad (7)$$

# 4 Experiments

## 4.1 Research Questions

In this section, we design extensive experiments to validate whether the proposed method obtains high-quality uncertainty estimation. Concretely, the following four research questions will be investigated.

**RQ1:** Whether E-NER improves the quality of confidence estimation in contrast to prior work?

| Dataset | Sentences | Types | Domain |
|---|---|---|---|
| CoNLL2003 | 22,137 | 4 | Newswire |
| OntoNotes 5.0 | 76,714 | 18 | General |
| WikiGold | 1,696 | 4 | General |

Table 1: Statistics of the NER dataset.

| Dataset | Sentences | Entities | OOV Rate |
|---|---|---|---|
| TwitterNER | 3257 | 3990 | 0.62 |
| CoNLL2003-Typos | 2676 | 4130 | 0.71 |
| CoNLL2003-OOV | 3685 | 5648 | 0.96 |

Table 2: Statistics of OOV entities in the test set.

**RQ2:** Can uncertainty provided by E-NER achieve better OOV/OOD detection performance?

**RQ3:** Can E-NER improve the model generalization ability on OOV samples?

**RQ4:** Can E-NER help to find valuable instances to improve the sample efficiency of NER model training?

Following these four research questions, we provide further discussions on our method including ablation studies and limitations.

## 4.2 Datasets and Metrics

**Datasets from Different Domains**. To answer the above research questions, we choose three widely-used datasets, including CoNLL2003 (Tjong Kim Sang and De Meulder, 2003), OntoNotes 5.0 (Weischedel et al., 2013)[2] and WikiGold (Balasuriya et al., 2009). The statistics are displayed in Table 1.

**OOV Datasets**. We further choose three public OOV datasets, including TwitterNER (Zhang et al., 2018), CoNLL2003-Typos (Wang et al., 2021), and CoNLL2003-OOV (Wang et al., 2021). The statistics are displayed in Table 2.

**Metrics**. We evaluate the results using three metrics: F1, Expected Calibration Error (ECE), and Area Under the ROC Curve (AUC). F1 is a commonly used performance indicator in NER. ECE is a metric that measures the confidence calibration of a model, with a low score indicating a well-calibrated model. AUC is a commonly used metric for evaluating the performance of binary classifiers, and we use it to evaluate the OOV/OOD detection performance. Their detailed computations are described in the Appendix §C.2.

[2] https://catalog.ldc.upenn.edu/LDC2013T19

| Setting | Typos | | OOV | | OOD | |
|---|---|---|---|---|---|---|
| | Con | Unc | Con | Unc | Con | Unc |
| BERT-Tagger (Devlin et al., 2019) | 0.812 | 0.812 | 0.689 | 0.751 | 0.674 | 0.756 |
| -EDL | 0.805 | 0.808 | 0.699 | 0.759 | 0.693 | 0.767 |
| -E-NER(ours) | **0.820** | **0.817** | **0.700** | **0.760** | **0.769** | **0.799** |
| SpanNER(Fu et al., 2021) | 0.717 | 0.783 | 0.614 | 0.773 | 0.623 | 0.799 |
| -EDL | 0.701 | 0.759 | 0.607 | 0.760 | 0.620 | 0.792 |
| -E-NER(ours) | **0.741** | **0.792** | **0.640** | **0.796** | **0.676** | **0.824** |
| Seq2Seq (Yan et al., 2021) | 0.825 | 0.833 | 0.724 | 0.794 | 0.797 | 0.820 |
| -EDL | **0.829** | 0.830 | 0.729 | 0.787 | 0.793 | 0.818 |
| -E-NER(ours) | 0.824 | **0.841** | **0.743** | **0.803** | **0.822** | **0.847** |

Table 3: Evaluation results of OOV/OOD detection in terms of AUC. The three binary detection tasks can use either confidence (Con) or uncertainty (Unc) for classification.

| Setting | CoNLL2003 | | OntoNotes 5.0 | |
|---|---|---|---|---|
| | F1(↑) | ECE(↓) | F1(↑) | ECE(↓) |
| BERT-Tagger | 91.32 | 0.0845 | 88.20 | 0.1053 |
| -EDL | 91.36 | 0.0755 | 88.09 | 0.0838 |
| -E-NER(ours) | **91.55** | **0.0739** | **88.74** | **0.0603** |
| SpanNER | 91.94 | 0.0673 | 87.82 | 0.0609 |
| -EDL | 91.97 | 0.0481 | 87.39 | 0.0474 |
| -E-NER(ours) | **92.06** | **0.0414** | **88.44** | **0.0434** |
| Seq2Seq | 93.05 | 0.0324 | 89.89 | 0.0375 |
| -EDL | 92.84 | 0.0322 | 90.22 | 0.0329 |
| -E-NER(ours) | **93.15** | **0.0225** | **90.64** | **0.0328** |

Table 4: Evaluation results in various NER systems, in terms of F1 (%) and ECE for evaluating performance and confidence quality, respectively.

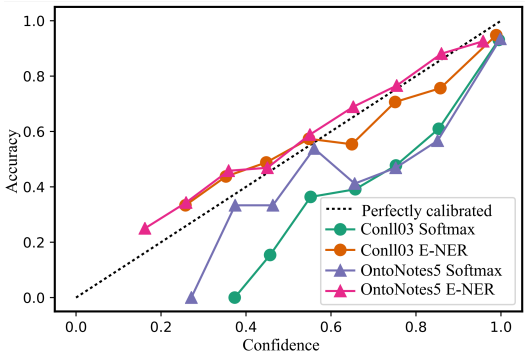



Figure 4: Model calibration curve. The basic encoder is SpanNER. This figure is depicted by evaluating subsets separately, where each subset has the same range of confidence.

### 4.3 Experiment Setting

We conduct experiments on three popular NER paradigms: sequence labeling, span-based, and Seq2Seq. The following three models are chosen for evaluating each paradigm.

**BERT-Tagger** (Devlin et al., 2019). It follows the classical paradigm, recognizing entities via *sequence labeling*.

**SpanNER**[3] (Fu et al., 2021). It enumerates all spans and detects entities from them. For simplicity, we use the original span-based method, without any constraints or data processing.

**Seq2Seq**[4] (Yan et al., 2021). It is a generative model based on BART, which does not require additional labeling strategies and entity enumeration.

In the experiments, all the reported results are the average of five runs. The experiment details are introduced in Appendix §C.

[3]https://github.com/neulab/spanner
[4]https://github.com/yhcc/BARTNER

### 4.4 Research Question Discussions

#### 4.4.1 Confidence Estimation Quality

To answer the first research question, an important concept should be clarified, i.e., *what is qualified confidence?* This concept should have a positive correlation with performance, meaning that higher confidence should indicate better performance and vice versa, as depicted by the dashed line in Figure 4. Our findings reveal that on both datasets, Softmax is far below the perfectly calibrated line, indicating that confidence does not reflect performance well, and it is an example of *over-confidence*. However, E-NER is found to approach the perfect calibrated line. This suggests that E-NER can produce well-qualified confidence.

We further evaluate all paradigms and present the results in Table 4. It can be observed that E-NER consistently performs the best across all paradigms. This demonstrates that E-NER can be effectively applied in various frameworks. When comparing EDL to the original models, it is observed that while EDL improves confidence estimation, it also

| Methods | TwitterNER | CoNLL2003 | |
|---|---|---|---|
| | | Typos | OOV |
| VaniIB (Alemi et al., 2017) | 71.19 | 83.49 | **70.12** |
| DataAug (Dai and Adel, 2020) | 73.69 | 81.73 | 69.60 |
| SpanNER (BERT large) | 71.57 | 81.83 | 64.43 |
| SpanNER (RoBERTa large) | 71.70 | 82.85 | 64.70 |
| SpanNER (AlBERT large) | 70.33 | 82.49 | 64.12 |
| EDL-SpanNER (BERT large) | 74.14 | 82.89 | 68.40 |
| E-SpanNER (BERT base) | 74.94 | 83.31 | 67.99 |
| E-SpanNER (BERT large) | **75.64** | **83.64** | 69.71 |
| Δ E-NER-NER vs. SpanNER | 4.07↑ | 1.81↑ | 5.28↑ |

Table 5: Evaluation results of generalization on OOV samples in terms of F1 (%). To compare fairly, we also choose SpanNER as the basic encoder.

results in a decline in performance. For example, on OntoNotes 5.0 dataset, EDL performs worse than BERT-Tagger and SpanNER in terms of the F1 metric. This highlights the limitations of directly applying the EDL approach. In contrast, E-NER performs the best on both metrics, demonstrating that it can provide better-qualified confidence without negatively impacting performance, and even achieving slight improvements in all settings. A typical reliability diagram is also included in Appendix §B.1 for a more detailed representation.

#### 4.4.2 OOV/OOD Detection

The typical usage of uncertainty is to detect whether an instance is OOV/OOD or not, as large uncertainty tends to reveal unnatural instances, such as OOV and OOD. To evaluate uncertainty from this usage (RQ2), we choose three binary detection tasks, including typos, OOV, and OOD. The results are shown in Table 3.

Firstly, it can be observed that, when compared to the original model of each paradigm, EDL does not improve the performances in most experiments of the three paradigms. This verifies that EDL is not effective in addressing the *OOV/OOD entity discrimination* challenge of NER. Then we found that E-NER significantly outperforms the original models and EDL in various paradigms. In particular, in span-based OOD detection, E-NER outperforms SpanNER by +5.3% and EDL by +5.6% on AUC when using confidence for detection. This demonstrates the effectiveness of E-NER in distinguishing whether an entity is OOV/OOD or not. Note that using uncertainty is better than using confidence for OOV/OOD detection in most cases.

| Setting | CoNLL2003 | | OntoNotes 5.0 | |
|---|---|---|---|---|
| | Ratio | F1(↑) | Ratio | F1(↑) |
| Random | 5.5% | 85.39 | 3.0% | 79.47 |
| Entropy | 5.5% | 88.29 | 3.0% | 84.80 |
| MC dropout | 5.5% | 88.67 | 3.0% | 86.06 |
| EDL | 5.5% | 90.51 | 3.0% | 86.25 |
| E-NER | 5.5% | **90.88** | 3.0% | **86.68** |

Table 6: Evluation results of in-domain data selection in terms of F1 (%). Ratio indicates the proportion of selected samples out of the whole training set.

| Setting | WikiGold$_{\leftarrow \text{CoNLL.}}$ | | CoNLL2003$_{\leftarrow \text{Onto.}}$ | |
|---|---|---|---|---|
| | Ratio | F1(↑) | Ratio | F1(↑) |
| Random | 4.8% | 53.67 | 4.7% | 84.23 |
| Entropy | 4.8% | 80.63 | 4.7% | 88.81 |
| MC dropout | 4.8% | 82.87 | 4.7% | 90.32 |
| EDL | 4.8% | 83.32 | 4.7% | 90.12 |
| E-NER | 4.8% | **84.08** | 4.7% | **90.52** |

Table 7: Evaluation results of cross-domain data selection in terms of F1 (%). The left side of the arrow ← is the target domain, and the right side is the source domain.

#### 4.4.3 Generalization on OOV Samples

Another benefit of well-qualified confidence is the robustness to noise, since the model is properly calibrated without over or under-confidence. Thus, we further investigate E-NER's generalizing ability on OOV samples (RQ3). The results on three OOV datasets are reported in Table 5.

It is first observed that E-NER (BERT large) achieves the best performances on TwitterNER and CoNLL2003-Typos datasets, and competitive performance on CoNLL2003-OOV. Compared with a strong baseline SpanNER (BERT large), E-NER (BERT large) significantly outperforms it by +4.07%, +1.81% and +5.28% on three datasets, respectively. This validates the generalizing ability of our approach. Secondly, by comparing EDL (BERT large) and E-NER s(BERT large), our method also achieves consistently better performances. This further validates that our proposed two uncertainty-guided loss terms effectively promote the robustness against OOV samples.

#### 4.4.4 Sample Efficiency

In active learning, a sample's uncertainty can be utilized for data selection. Then whether the selected samples are valuable also suggests the quality of uncertainty. To evaluate E-NER from this perspec-

| Setting | CoNLL2003 | | OntoNotes 5.0 | |
|---|---|---|---|---|
| | F1 | ECE | F1 | ECE |
| E-NER | 92.06 | **0.041** | **88.44** | **0.043** |
| -UNM | **92.10** | 0.058 | 88.21 | 0.051 |
| -IW | 91.95 | 0.045 | 87.77 | 0.042 |

Table 8: Evaluation results of ablation study in terms of F1 (%) and ECE.

tive (RQ4), we design in-domain and cross-domain sample selection experiments. The results are displayed in Table 6 and Table 7, respectively.

It is found that using the same scale of samples, E-NER achieves consistently the best performances in both the in-domain and cross-domain settings. This verifies that uncertainty predicted by E-NER has better quality. Concretely, MC dropout attains uncertainty with multiple runs of sub-models, which costs time and memory. Though outperforming naive random selection and entropy of softmax, MC dropout is still less performed than EDL and E-NER, which both directly compute the uncertainty in one forward pass. Then we see that EDL does not always outperform MC dropout, as the cross-domain experiment $\text{CoNLL2003}_{\leftarrow\text{Onto}}$ shown. Yet E-NER, concentrating on two issues of NER task, is universally effective, and can better handle the challenges of an open environment.

### 4.5 Further Analysis

**Ablation Study**. To explore the effects of individual loss terms, the ablation study is presented in Table 8. It is observed that removing each loss term would cause performance declines in most evaluation metrics. Concretely, removing IW causes the F1 score to decrease more than removing UNM. On the contrary, removing UNM makes a significant degradation in ECE. Overall, this study indicates that the proposed uncertainty-guided terms are both effective.

**Why E-NER Works**. We incorporate two uncertainty-guided loss terms into EDL. Firstly, IW is designed for sparse entities which leads to an imbalance problem. Using uncertainties as weights helps the model training to pay more attention to entities of interest. As reported in Table 8, IW is effective in improving the F1 score. Secondly, UNM is proposed to deal with OOV/OOD entities. Such entities should have larger uncertainties compared to normal ones, however, naive EDL does not model this explicitly. E-NER increases the uncertainty of mispredictions which are relatively close to OOV/OOD entities. As shown in Table 8, UNM helps to improve the quality of uncertainty estimation. These two uncertainty-guided loss terms target different NER issues, and using uncertainty (IW) and learning uncertainty (UNM) interactively allows E-NER to perform well in various experimental settings. Furthermore, we showcase actual predictions in Appendix §B.2.

## 5 Related Work

**NER Paradigm**. NER is a fundamental task in information extraction. The mainstream methods of NER can be divided into three categories: sequence labeling, span-based, and Seq2Seq. Sequence labeling methods assign a label to each token in a sentence to identify flat entities, and are better at handling longer entities with lower label consistency (Fu et al., 2021). Span-based methods, which enumerate and classify entity sets in a sentence according to the maximum span length, perform better on sentences with OOV words and entities of medium length (Alemi et al., 2017; Dai and Adel, 2020; Fu et al., 2021). Seq2Seq methods directly generate the entities and corresponding labels in the sentence, and are capable of handling various NER subtasks uniformly (Yan et al., 2021). Recently, NER systems are undergoing a paradigm shift (Akbik et al., 2018; Yan et al., 2019), using one paradigm to handle multiple types of NER tasks. Zhang et al. (2022) analysis the incorrect bias in Seq2Seq from the perspective of causality, and designed a data augmentation method based on the theory of backdoor adjustment, making Seq2Seq more suitable for unified NER tasks.

**Uncertainty Estimation**. Bayesian deep learning uses Bayesian principles to estimate uncertainty in DNN parameters. However, modeling uncertainty in network parameters does not guarantee accurate estimation of predictive uncertainty (Sensoy et al., 2021). Recently, there has been a trend in using the output of neural networks to estimate the parameters of the Dirichlet distribution for uncertainty estimation (Sensoy et al., 2018; Malinin and Gales, 2018). The EDL (Sensoy et al., 2018) has the advantages of generalizability and low computational cost, making it applicable to various tasks (Han et al., 2021; Hu and Khan, 2021). However, their uncertainty estimates have difficulty expressing uncertainties outside the domain (Amini et al., 2020; Hu and Khan, 2021). In contrast, the Prior

Networks (Malinin and Gales, 2018) require OOD data during training to distinguish in-distribution (ID) and OOD data. When the NER model encounters unseen entities (e.g., OOV and OOD), it is easy to make unreliable predictions, which are often considered from the perspective of data augmentation or information theory (Fukuda et al., 2020; Wang et al., 2022), but there is no guarantee that these methods will achieve a balance between performance and robustness.

## 6 Conclusion

In this work, we study the problem of trustworthy NER by leveraging evidential deep learning. To address the issues of *sparse entities* and *OOV/OOD entities*, we propose E-NER with two uncertainty-guided loss terms. Extensive experimental results demonstrate that the proposed method can be effectively applied to various NER paradigms. The uncertainty estimation quality of E-NER is improved without harming performance. Additionally, the well-qualified uncertainties contribute to detecting OOV/OOD, generalization, and sample selection. These results validate the superiority of E-NER on real-world problems.

## Limitations

Our work is the first attempt to explore how evidential deep learning can be used to improve the reliability of current NER models. Despite the improved performance and robustness, our work has limitations that may guide our future work.

First, we propose a simple method to treat hard samples (such as outliers) in the dataset as OOV/OOD samples, enabling the model to detect OOV/OOD data with minimal cost. However, there is still a certain gap between these hard samples and the real OOV/OOD data. OOV/OOD detection performance can still be improved by further incorporating more real OOV/OOD samples, for example, real OOD data from other domains, well-designed adversarial examples, generated OOV samples by data augmentation techniques, etc.

Second, we evaluate the versatility of E-NER by applying it to mainstream NER paradigms. However, there are still other paradigms, such as Hypergraph-based methods (Lu and Roth, 2015) and the W$^2$NER (Li et al., 2022) approach in recent work, that could be evaluated in the future.

## Acknowledgements

We sincerely thank all the anonymous reviewers for providing valuable feedback. This work is supported by the youth program of National Science Fund of Tianjin, China (Grant No. 22JCQNJC01340), the Fundamental Research Funds for the Central University, Nankai University (Grant No. 63221028), and the key program of National Science Fund of Tianjin, China (Grant No. 21JCZDJC00130)

## 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 (COLING)*, pages 1638–1649.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *International Conference on Learning Representations (ICLR)*, pages 1–19.

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. 2020. Deep evidential regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14927–14937.

Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R. Curran. 2009. Named entity recognition in Wikipedia. In *Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources (People's Web)*, pages 10–18.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural network. In *International conference on machine learning (ICML)*, pages 1613–1622.

Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. 2020. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1356–1367.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*, pages 3861–3867.

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 (NAACL)*, pages 4171–4186.

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 (ACL-IJCNLP)*, pages 7183–7195.

Nobukazu Fukuda, Naoki Yoshinaga, and Masaru Kitsuregawa. 2020. Robust Backed-off Estimation of Out-of-Vocabulary Embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4827–4838.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning (ICML)*, pages 1050–1059.

Alex Graves. 2011. Practical variational inference for neural networks. In *Advances in neural information processing systems (NeurIPS)*, page 2348–2356.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1321–1330.

Zongbo Han, Changqing Zhang, Huazhu Fu, and Joey Tianyi Zhou. 2021. Trusted multi-view classification. In *International Conference on Learning Representations (ICLR)*, pages 1–16.

Yibo Hu and Latifur Khan. 2021. Uncertainty-aware reliable text classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (SIGKDD)*, pages 628–636.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, pages 1–15.

Durk P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems (NeurIPS)*, pages 2575–2583.

Anna-Kathrin Kopetzki, Bertrand Charpentier, Daniel Zügner, Sandhya Giri, and Stephan Günnemann. 2021. Evaluating robustness of predictive uncertainty estimation: Are dirichlet-based models reliable? In *International Conference on Machine Learning (ICML)*, pages 5707–5718.

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 (NAACL)*, pages 260–270.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. 2018. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations (ICLR)*, pages 1–16.

Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022. Unified named entity recognition as word-word relation classification. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*, pages 10965–10973.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, pages 1–18.

Wei Lu and Dan Roth. 2015. Joint mention extraction and classification with mention hypergraphs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 857–867.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *Advances in neural information processing systems (NeurIPS)*, page 7047–7058.

Francesco Pinto, Philip HS Torr, and Puneet K Dokania. 2022. An impartial take to the cnn vs transformer robustness contest. In *European Conference on Computer Vision (ECCV)*, pages 466–480.

Murat Sensoy, Lance M. Kaplan, and Melih Kandemir. 2018. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems (NeurIPS)*, page 3183–3193.

Murat Sensoy, Maryam Saleki, Simon Julier, Reyhan Aydogan, and John Reid. 2021. Misclassification risk and uncertainty quantification in deep classifiers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2484–2492.

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 (HLT-NAACL)*, pages 142–147.

Xiao Wang, Shihan Dou, Limao Xiong, Yicheng Zou, Qi Zhang, Tao Gui, Liang Qiao, Zhanzhan Cheng, and Xuanjing Huang. 2022. MINER: Improving out-of-vocabulary named entity recognition from an information theoretic perspective. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*, pages 5590–5600.

Xiao Wang, Qin Liu, Tao Gui, Qi Zhang, Yicheng Zou, Xin Zhou, Jiacheng Ye, Yongxin Zhang, Rui Zheng,

and Zexiong and Pang. 2021. TextFlint: Unified multilingual robustness evaluation toolkit for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations (ACL-IJCNLP)*, pages 347–355.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes Release 5.0. In *3*. Abacus Data Network.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6442–6454.

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. TENER: adapting transformer encoder for named entity recognition. *CoRR*, abs/1911.04474.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various NER subtasks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 5808–5822.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 6470–6476.

Qi Zhang, Jinlan Fu, Xiaoyu Liu, and Xuanjing Huang. 2018. Adaptive co-attention network for named entity recognition in tweets. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, page 5674–5681.

Shuai Zhang, Yongliang Shen, Zeqi Tan, Yiquan Wu, and Weiming Lu. 2022. De-bias for generative extraction in unified NER task. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (ACL)*, pages 808–818.

Enwei Zhu and Jinpeng Li. 2022. Boundary smoothing for named entity recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7096–7108.

| | BERT-Tagger | SpanNER | Seq2Seq |
|---|---|---|---|
| Input | $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ | $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ | $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ |
| Processing | - | Enumerate all spans<br>$S = \{s^{(1)}, s^{(2)}, ..., s^{(m)}\}$ | Obtain start and end indexes of entities<br>$Y = \{y_1^b, y_1^e, y_1, ..., y_k^b, y_k^e, y_k\}$ |
| Hidden state | $h = \text{Encoder}(X)$;<br>$h \in \mathbb{R}^{n \times d}$ | $h = \text{Encoder}(s^{(i)})$;<br>$h \in \mathbb{R}^{d}$ | $h_t = \text{EncoderDecoder}(X, Y_{<t})$;<br>$h_t \in \mathbb{R}^{d}$ |
| Inference | Token-level classification | Span-level classification | Target sequence $Y$ generation |

Table 9: Explanation of the three NER paradigms.

## A NER Paradigms

Here we introduce three popular NER paradigms, shown in Table 9.

**BERT-Tagger**. It follows the sequence labeling paradigm, which aims to assign a tagging label $Y = \{y^{(1)}, ..., y^{(n)}\}$ to each word in a sequence $X = \{x^{(1)}, ..., x^{(n)}\}$. We use BERT-Tagger (Devlin et al., 2019) as the baseline method for sequence labeling. The labeling method adopts a BIO tag set, which indicates the beginning and interior of an entity, or other words. $X$ is fed to BERT to obtain hidden states, followed by a nonlinear classifier to classify each word.

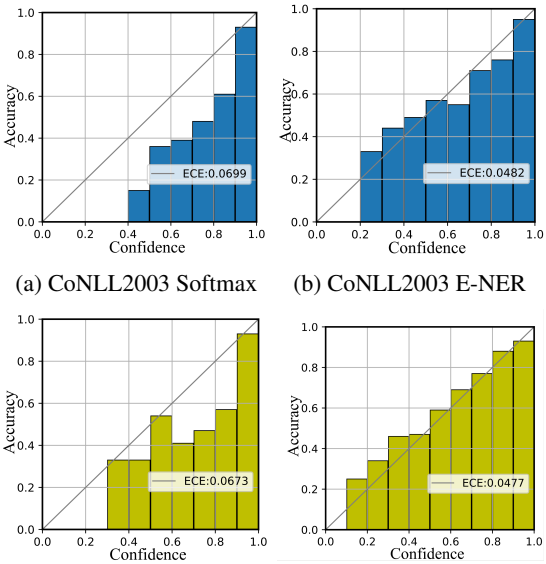
**SpanNER**. Given an input sentence $X = \{x^1, ..., x^n\}$, SpanNER enumerates all spans and obtains a set $S = \{s^{(1)}, ..., s^{(i)}, ..., s^{(m)}\}$. Then it assigns each span an entity label $y$ (Fu et al., 2021). The maximum length $l$ of the span is artificially set. Assume a sentence's length is $n$ and the maximum span length is set to 2, the subscript of the span set can be expressed as $\{(1,1),(1,2)...(n-1,n-1),(n-1,n),(n,n)\}$. Each span is fed into the encoder to obtain a vector representation.

**Seq2Seq**. As presented in Table 9, given an input sentence $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$, the target sequence is represented as $Y = \{y_1^b, y_1^e, y_1, ..., y_k^b, y_k^e, y_k\}$. This target sequence indicates $X$ describes $k$ entities. Take the first entity as an example, its beginning and end indexes are $y_1^b$ and $y_1^e$, with entity category $y_1$. This method learns in a sequence-to-sequence manner (Yan et al., 2021).

## B Additional Experimental Analysis

### B.1 Reliability Diagrams

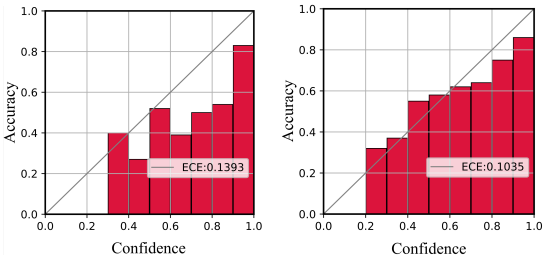
We further depict the reliability diagrams to evaluate the quality of uncertainty estimation. As shown in Figure 5 and Figure 6, the confidence range is equally divided into ten bins. Then the subset within the same confidence range is utilized to compute the accuracy.




(a) CoNLL2003 Softmax (b) CoNLL2003 E-NER

(c) OntoNotes 5.0 Softmax (d) OntoNotes 5.0 E-NER

Figure 5: Reliability diagrams.




(a) OOD Softmax (b) OOD E-NER

Figure 6: Reliability diagrams of OOD entities. CoNLL2003 is used as the training set. The testing set of WikiGold is used for evaluating the OOD samples.

As shown in Figure 5, the confidence of Softmax represents poor accuracy, indicating it is overconfident. Then compared with Softmax, E-NER nearly approaches the perfectly calibrated line and

| Case | Sentence | Softmax+Entropy | E-NER |
|---|---|---|---|
| * | Mapping: {MIS: miscellaneous; PER: person; ORG: organization; O: non-entity} | Entity: {Predcition; Confidence%; Uncertainty%} | |
| $\text{I}_{ID}$ | A visit to the computer centre offering $Internet^{\text{E}^1}_{[\text{MIS}]}$ services found a $European^{\text{E}^2}_{[\text{MIS}]}$ official clicking away on his mouse. | $\text{E}^1$ {O ; 99.9 ; 8.0} $\text{E}^2$ {MIS ; 99.9 ; 3.0} | $\text{E}^1$ {O ; 42.0 ; 70.8} $\text{E}^2$ {MIS ; 92.7 ; 8.9} |
| $\text{II}_{ID}$ | $Lazio^{\text{E}^1}_{[\text{ORG}]}$ have injury doubts about striker $Pierluigi\ Casiragh^{\text{E}^2}_{[\text{PER}]}$. | $\text{E}^1$ {O ; 98.8 ; 7.3} $\text{E}^2$ {PER ; 99.9 ; 0.4} | $\text{E}^1$ {ORG ; 88.9 ; 12.5} $\text{E}^2$ {PER ; 98.3 ; 2.3} |
| $\text{III}_{OOV}$ | But the $Inthrnet^{\text{E}^1}_{[\text{MIS}]}$ , a global computer network. | $\text{E}^1$ {O ; 90.5 ; 23.1} | $\text{E}^1$ {MIS ;28.1 ; 70.0} |
| $\text{IV}_{OOD}$ | Redesignated 65 $Fighter\ Wing^{\text{E}^1}_{[\text{ORG}]}$ on 24 July 1943. | $\text{E}^1$ {O ; 99.2 ; 4.6} | $\text{E}^1$ {O ; 51.3 ; 60.7} |

Table 10: Case study of Softmax and E-NER under the span-based paradigm. The entities and their categories are already denoted in four sentences. The predicted entities with confidence (%) and uncertainty (%) scores are also presented. Incorrectly predicted entities are denoted by "Red E", whereas "Blue E" represents correctly predicted entities.

has a much smaller ECE score. This suggests that E-NER can yield well-qualified confidence, showing it is more trustworthy. Then the observations in Figure 6 are similar, which demonstrates the reliability of the proposed approach for OOD entities.

### B.2 Case Study

As presented in Table 10, we conduct a case study by choosing four typical cases, including ID, OOV, and OOD samples. The uncertainty of Softmax is computed with entropy.

The first case contains two MIS entities. Softmax and E-NER both wrongly predict the first entity to O category, with confidence scores of 99.9% and 42.0%, respectively. This shows that Softmax is over-confident even for error results. Yet E-NER can output a larger uncertainty score, suggesting unsure towards the prediction. Then the second case describes two entities. Softmax wrongly predicts the first ORG entity to O with large confidence, i.e. 98.8%. But E-NER can correctly detect the entity category as ORG.

Moreover, *Inthrnet* in the third sentence is a MIS entity, which is OOV due to misspelling. Softmax detects it as O with a confidence score of 90.5%, showing over-confident for errors. On the contrary, E-NER assigns a large uncertainty score for the OOV sample and correctly predicts the entity category. Similarly, the last case describes an OOD entity. It can be observed that E-NER outputs a large uncertainty score compared with Softmax.

Based on the cases and observations, we draw the following conclusions: 1) Softmax is over-confident, even for error prediction, OOV and OOD samples; 2) E-NER can recognize entities accurately and yield well-qualified uncertainties towards error, OOV and OOD samples. This contributes to the reliability and robustness of E-NER.

## C Implementation Details

### C.1 Model Parameters

In this paper, we implement three NER methods, including BERT-Tagger, SpanNER and Seq2Seq. The testing set is evaluated by the best model chosen by the development set. The implementation details are shown as follows.

**BERT-Tagger**. BERT-Tagger[5] adopts BERT-large-cased as the base encoder (Devlin et al., 2019). We set the dropout rate as 0.2, the training batch size as 16, and the weight decay as 0.02. All models in this paradigm use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 2e-5. Sentences are truncated to a maximum length of 256. The initial value for $\lambda_0$ is set to 1e-02.

**SpanNER**. Following the original SpanNER[6] (Fu et al., 2021), we adopt BERT-large-uncased as the base encoder (Devlin et al., 2019). The dropout rate is set to 0.2. All models in this paradigm are trained using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 1e-5, with the training batch size as 10. To improve training efficiency, sentences are truncated to a maximum length of 128, and the maximum length of span enumeration is set to 4. The sampling times for MC dropout are set to 5 in the experiments. The

[5]https://github.com/google-research/bert
[6]https://github.com/neulab/spanner.

initial value of $\lambda_0$ is set to 1e-02. We use heuristic decoding and retain the highest probability span for flattened entity recognition in span-based methods.

**Seq2Seq**. Following Yan et al. (2021), we exploit BART-Large model[7]. BART model is fine-tuned with the slanted triangular learning rate warmup. The warmup step is set to 0.01. The training batch size is set to 16. The initial value of $\lambda_0$ is set to 1e-3.

### C.2 Evaluation Metrics

**ECE**. It denotes the expected calibration error, which aims to evaluate the expected difference between model prediction confidence and accuracy (Guo et al., 2017). Figure 6 depicts the difference in a geometric manner. The concrete formulation is as follows:

$$\mathrm{ECE} = \sum_{i=1}^{|B|} \frac{N_i}{N} |\mathrm{acc}(b_i) - \mathrm{conf}(b_i)|, \tag{8}$$

where $b_i$ represents the $i$-th bin and $|B|$ represents the total number of bins, setting to 10 in our experiment. $N$ denotes the number of total samples. $N_i$ represents the number of samples in the $i$-th bin. $\mathrm{acc}(b_i)$ denotes the accuracy and $\mathrm{conf}(b_i)$ denotes the average of confidences in the $i$-th bin.

**AUC**. The area under the curve (AUC)[8] is a commonly used metric for evaluating the performance of binary classifiers. The formulation is as follows:

$$\mathrm{AUC}(f) = \frac{\sum_{t_0 \in \mathcal{D}^0} \sum_{t_1 \in \mathcal{D}^1} \mathbf{1}[f(t_0) < f(t_1)]}{|\mathcal{D}^0| \cdot |\mathcal{D}^1|} \tag{9}$$

where $\mathcal{D}^0$ is the set of negative examples, and $\mathcal{D}^1$ is the set of positive examples. $\mathbf{1}[f(t_0) < f(t_1)]$ denotes an indicator function which returns 1 if $f(t_0) < f(t_1)$ otherwise return 0.

In this paper, we evaluate the performance of OOV/OOD detection using the AUC metric. Specifically, we consider two settings for the AUC score:

- **Con**. It uses confidence as a classifier. The correct entity recognition is a positive example $\mathcal{D}^1$, and the entity recognition error is a negative example $\mathcal{D}^0$.
- **Unc**. It uses uncertainty as a classifier. Wrong prediction results of OOV/OOD entities are considered positive examples, denoted as $\mathcal{D}^1$. Correct prediction results of in-domain entities are considered negative examples, recorded as $\mathcal{D}^0$. These metrics assess the classifier's capability in detecting OOV/OOD entities.

### C.3 EDL Optimization Function

In this section, we give a detailed formulation of the EDL optimization function. Eq. 1 introduces the density of the Dirichlet distribution. As the classification loss item of EDL, its cross-entropy loss function is as follows:

$$\begin{aligned}\mathcal{L}_{CLS}^{(i)} &= \frac{\int \left[\sum_{c=1}^{C} -y_c^{(i)} \log(p_c^{(i)})\right]}{B(\boldsymbol{\alpha}^{(i)})} \prod_{c=1}^{C} p_c^{\alpha_c^{(i)}-1} d\mathbf{p}^{(i)} \\ &= \sum_{c=1}^{C} y_c^{(i)} \left(\psi(S^{(i)}) - \psi(\alpha_c^{(i)})\right).\end{aligned} \tag{10}$$

The KL divergence calculation function under the Dirichlet distribution takes the following form and serves as the category penalty term in EDL:

$$\begin{aligned}\mathcal{L}_{KL}^{(i)} &= KL[\mathrm{Dir}(\mathbf{p}^{(i)}|\widetilde{\boldsymbol{\alpha}}^{(i)})||\mathrm{Dir}(\mathbf{p}^{(i)}|\mathbf{1})] \\ &= \log\left(\frac{\Gamma(\sum_{c=1}^{C} \widetilde{\alpha}_c^{(i)})}{\Gamma(C)\prod_{c=1}^{C}\Gamma(\widetilde{\alpha}_c^{(i)})}\right) \\ &+ \sum_{c=1}^{C}(\widetilde{\alpha}_c^{(i)} - 1)\left[\psi(S^{(i)}) - \psi(\sum_{j=1}^{C}\widetilde{\alpha}_j^{(i)})\right].\end{aligned} \tag{11}$$

Finally, we get the loss function for overall EDL learning:

$$\mathcal{L}_{EDL} = \sum_{i=1}^{N} (\mathcal{L}_{CLS}^{(i)} + \mathcal{L}_{KL}^{(i)}) \tag{12}$$

[7] https://github.com/yhcc/BARTNER

[8] sklearn.metrics.auc.html.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Section §*Limitations*

☐ A2. Did you discuss any potential risks of your work?
*Not applicable. Left blank.*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Abstract and Section §1*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

### B ☑ Did you use or create scientific artifacts?

*Section §2 and Section §4*

☑ B1. Did you cite the creators of artifacts you used?
*Section §2 , Section §4 and Section §6*

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*Section §4*

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Not applicable. Left blank.*

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Not applicable. Left blank.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*Not applicable. Left blank.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Section §4.2*

### C ☑ Did you run computational experiments?

*Section §4*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Section §4 and Section §C.1*

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Section §4 and Section §C.1*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Section §4 and Section §C.1*

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*Section §4 and Section §C.1*

**D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?**

*Left blank.*

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*No response.*

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*No response.*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*No response.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*No response.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*No response.*