A Confidence-based Partial Label Learning Model for Crowd-Annotated Named Entity Recognition

Limao Xiong[1], **Jie Zhou**[1]*, **Qunxi Zhu**[2], **Xiao Wang**[1], **Yuanbin Wu**[3], **Qi Zhang**[1], **Tao Gui**[4], **Xuanjing Huang**[1], **Jin Ma**[5], **Ying Shan**[5]

[1] School of Computer Science, Fudan Univerisity

[2] Research Institute of Intelligent Complex Systems, Fudan University

[3] The Department of Computer Science and Technology, East China Normal University

[4] Institute of Modern Languages and Linguistics, Fudan University

[5] Applied Research Center (ARC), Tencent PCG

Abstract

Existing models for named entity recognition (NER) are mainly based on large-scale labeled datasets, which always obtain using crowdsourcing. However, it is hard to obtain a unified and correct label via majority voting from multiple annotators for NER due to the large labeling space and complexity of this task. To address this problem, we aim to utilize the original multi-annotator labels directly. Particularly, we propose a Confidence-based Partial Label Learning (CPLL) method to integrate the prior confidence (given by annotators) and posterior confidences (learned by models) for crowd-annotated NER. This model learns a token- and content-dependent confidence via an Expectation–Maximization (EM) algorithm by minimizing empirical risk. The true posterior estimator and confidence estimator perform iteratively to update the true posterior and confidence respectively. We conduct extensive experimental results on both real-world and synthetic datasets, which show that our model can improve performance effectively compared with strong baselines.

1 Introduction

Named entity recognition (NER) plays a fundamental role in many downstream natural language processing (NLP) tasks, such as relation extraction (Bach and Badaskar, 2007), event extraction (Wadden et al., 2019; Zhou et al., 2022). Recently, by leveraging deep learning models, existing NER systems have witnessed superior performances on NER datasets. However, these models typically require a massive amount of labeled training data, such as MSRA (Levow, 2006), Ontonotes 4.0 (Weischedel et al., 2011), and Resume (Zhang and Yang, 2018). In real applications, we often need to consider new types of entities in new domains where we do not have existing annotated. The majority way to label the data at a lower cost is crowdsourcing (Peng and Dredze, 2015), which labels the data using multiple annotators.

The crowd-annotated datasets are always low quality for the following two reasons. First, as an exchange, crowd annotations are always nonexperts. Various annotators may have different interpretations of labeling guidelines. Moreover, they may make mistakes in the labeling process. It is hard to require a number of annotations to reach an agreement. For example, annotator 1 labels “David and Jack" as a PER entity, while the correct label is “David" and “Jack" under our guidelines (Table 1). Also we should label the continuous time and place as one entity (e.g, “tomorrow at 10:00 a.m." and “company (room 1003)"). Second, due to the ambiguous word boundaries and complex composition, the NER task is more challenging compared with the text classification tasks. Annotator 3 ignores the token “a.m." for the time entity and adds “the" as part of the place entity falsely. Also, he/she misses the person entities in the text. In this paper, we focus on building a powerful NER system based on crowd-annotated data, which is of low quality.

There are two main ways to utilize crowd-annotated data. One simple and important way to obtain high-quality annotations for each input instance is majority voting. As shown in Table 1, the majority voting method can not obtain the correct answers from these three annotations well. The right labels (e.g., “David", “Jack", “tomorrow at 10:00 a.m.", and “company (room 1003)") are only annotated by annotators 1 and 2 once. Another majority of work models the differences among annotators by finding the trustworthy annotators (Rodrigues et al., 2014; Nguyen et al., 2017; Yang et al., 2018). From Table 1, we can find that none of the three annotators labels the entities absolutely right. Thus, these two kinds of methods are a waste of human labor.

To address this problem, we translated this task into a partial label learning (PLL) problem, which

* Corresponding author, jie_zhou@fudan.edu.cn.

Findings of the Association for Computational Linguistics: ACL 2023, pages 1375–1386
July 9-14, 2023

Annotator 1		PER ✗					TIME ✓				PLACE ✓		
		David and Jack	will	hold	a	meeting	tomorrow at 10:00 a.m.	in	the	company	(room 1003)	.	
Annotator 2	PER ✓		PER ✓				TIME ✗		TIME ✗		PLACE ✗		PLACE ✗
	David	and	Jack	will	hold	a meeting	tomorrow	at	10:00 a.m.	in the	company	(	room 1003).
Annotator 3							TIME ✗				PLACE ✗		
	David	and	Jack	will	hold	a meeting	tomorrow at 10:00	a.m.	in		the company (room 1003	).	

Table 1: The spans marked with blue, green, and red are time (TIME), person (PER), and place (PLACE) entities labeled by three annotators.

trains the model based on the dataset where each sample is assigned with a set of candidate labels (Cour et al., 2011; Wen et al., 2021). Thus, it is natural to utilize all human labor via PLL, which can be divided into two types: 1) average-based methods which consider each candidate class equally (Hüllermeier and Beringer, 2006; Zhang and Yu, 2015); 2) identification-based methods which predict the ground-truth label as a latent variable via a translation matrix to describe the scores of each candidate label (Feng and An, 2019; Yan and Guo, 2020; Feng et al., 2020). Despite extensive studies on PLL methods, there are still two challenges in our condition. One challenge (**C1**) is that these methods are criticized when the same candidate label occurs more than once. The general PLL is under the assumption that each candidate label is only been assigned once, while each sample may be assigned the same classes multiple times by the different annotators in our situation. Another challenge (**C2**) is that most of the existing studies about PLL focus on image or text classification tasks, while we focus on a more complex task, sequence labeling, where each token is asserted with a label. Thus, the token itself and its content should be considered in this task.

In this paper, we propose a Confidence-based Partial Label Learning (CPLL) model for crowd-annotated NER. For **C1**, we treat the classes' labeled number for each sample as prior confidence provided by the annotators. Also, we learn the confidence scores via an Expectation–Maximization (EM) algorithm (Dempster et al., 1977). We estimate the real conditional probability $P(Y = y|T = t, X = \mathbf{x})$ via a true posterior estimator based on the confidence that consists of the prior and posterior confidences. For **C2**, we learn a token- and content-dependent confidence via a confidence estimator to consider both the token t and sequence input $\mathbf{x}$, because the candidate labels are always token-dependent and content-dependent. In fact, our model can be applied to all the sequence labeling tasks, such as word segment, part of speech, etc. We conduct a series of experiments on one real-world dataset and four synthetic datasets. The empirical results show that our model can make use of the crowd-annotated data effectively. We also explore the influence of annotation inconsistency and balance of prior and posterior confidences.

The main contributions of this work are listed as follows.

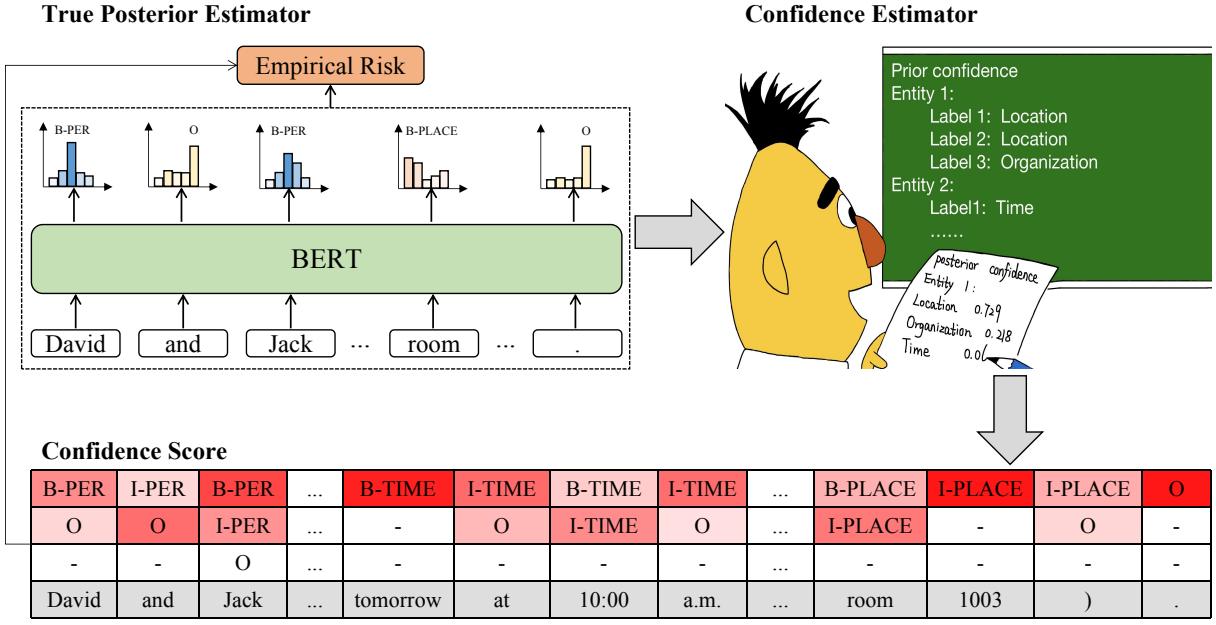
- To better utilize the crowd-annotated data, we propose a CPLL algorithm to incorporate the prior and posterior confidences for sequence labeling task (i.e., NER).
- To take the confidence scores into account, we design a true posterior estimator and confidence estimator to update the probability distribution of ground truth and token- and content-dependent confidence iteratively via the EM algorithm.
- Extensive experiments on both real-world and synthetic datasets show that our CPLL model outperforms the state-of-the-art baselines, which indicates that our model disambiguates the noise labels effectively.

2 Our Approach

In this section, we first give the formal definition of our task. Then, we provide an overview of our proposed CPLL model. Finally, we introduce the main components contained in our model.

2.1 Formal Definition

Given a training corpus $\mathcal{D} = \{\mathbf{x}_i, (\hat{Y}_i, A_i)\}_{i=1}^{|\mathcal{D}|}$, where $\mathbf{x} = \{t_1, t_2, ..., t_{|\mathbf{x}|}\}$, $(\hat{Y}, A) = \{(\hat{\mathbf{y}}_1, \mathbf{a}_1), (\hat{\mathbf{y}}_2, \mathbf{a}_2), ..., (\hat{\mathbf{y}}_{|\mathbf{x}|}, \mathbf{a}_{|\mathbf{x}|})\}$. Here, $\hat{\mathbf{y}} = \{y_1, y_2, ..., y_{|\hat{\mathbf{y}}|}\}$ is the candidate label set of the token t and $\mathbf{a} = [a_1, a_2, ..., a_{|\hat{\mathbf{y}}|}]$ is



B-PER	I-PER	B-PER	...	B-TIME	I-TIME	B-TIME	I-TIME	...	B-PLACE	I-PLACE	I-PLACE	O
O	O	I-PER	...	-	O	I-TIME	O	...	I-PLACE	-	O	-
-	-	O	...	-	-	-	-	...	-	-	-	-
David	and	Jack	...	tomorrow	at	10:00	a.m.	...	room	1003	)	.

Figure 1: The framework of our CPLL model, which consists of a true posterior estimator and confidence estimator. The true posterior estimator is used to predict the true posterior $P(Y = y|T = t, X = \mathbf{x})$ based on the confidence score learned by the confidence estimator. The confidence estimator learns the confidence based on the prior confidence obtained from annotators and the posterior confidence learned by the model.

labeled times obtained from annotations. Specifically, a is the labeled times of candidate label y for token t. $\hat{\mathbf{y}} \in \{2^{\mathcal{Y}} \backslash \varnothing \backslash \mathcal{Y}\}$ where $\mathcal{Y}$ is the label space and $2^{\mathcal{Y}}$ means the power set. For the rest of this paper, y denotes the true label of token t in text $\mathbf{x}$ unless otherwise specified. The goal of this task is to predict the truth posterior probability $P(Y = y|T = t, X = \mathbf{x})$ of token t in text $\mathbf{x}$.

2.2 Overview

In this paper, we propose a CONfidence-based partial Label Learning (CPLL) model for crowd-annotated NER (Figure 1). Particularly, we learn the true posterior $P(Y = y|T = t, X = \mathbf{x})$ via a true posterior estimator f and a confidence score $g(y; \hat{Y}, t, \mathbf{x})$ by minimizing the following risk.

$$R = \mathbb{E}_{p(\mathbf{x},\hat{\mathbf{y}})}\left[\sum_{t\in\mathbf{x}}\sum_{y}\underbrace{g(y;\hat{\mathbf{y}},t,\mathbf{x})}_{\text{Confidence}} * \underbrace{\mathcal{L}(f(y;t,\mathbf{x}),y)}_{\text{True posterior}}\right] \quad (1)$$

where the classifier $f(y; t, \mathbf{x})$ is used to predict $P(Y = y|T = t, X = \mathbf{x})$ and $\mathcal{L}$ is the loss. Particularly, we rely on the Expectation-Maximization algorithm (Dempster et al., 1977) to find the maximum likelihood parameters of CPLL by regarding the ground truth as a latent variable. In the M-step, we train a naive classifier f to predict the true posterior $P(Y = y|T = t, X = \mathbf{x})$ via a true posterior estimator (Section 2.3). In the E-step, we update the confidence score via a confidence estimator (Section 2.4), which consists of the prior confidences (calculated from annotations) and posterior confidences (learned by model).

2.3 True Posterior Estimator

First, we train a naive classifier as our true posterior estimator f to infer the true posterior $P(Y = y|T = t, X = \mathbf{x})$. To model the sequence, we adopt a pre-trained language model (BERT (Kenton and Toutanova, 2019)) $\mathcal{M}$ to learn a content-aware token representation. Specifically, we input the sequence $\mathbf{x} = \{t_1, t_2, ..., t_{|\mathbf{x}|}\}$ into $\mathcal{M}$ to obtain the sequence representations,

$$H = \mathcal{M}(\mathbf{x}, \theta_{\mathcal{M}}) \quad (2)$$

where $\theta_{\mathcal{M}}$ is the parameters of $\mathcal{M}$, $H = [h_1, h_2, ..., h_{|\mathbf{x}|}]$. h is token t's content-aware representation.

Then, we utilize a fully connected layer (FC) to predict the probability distribution,

$$f(y; t, \mathbf{x}) = \sigma(W * h + b) \quad (3)$$

where σ is a sigmoid function, $\theta_{FC} = \{W, b\}$ is the learnable parameters of FC. We regard $\theta = \{\theta_{\mathcal{M}}, \theta_{FC}\}$ as a parameter set of true posterior estimator f. Negative learning (Kim et al., 2019) is adopted, which not only considers "the token

belongs to positive label (candidate label $y \in \hat{\mathbf{y}}$)" but also "the token does not belong to negative label (its complementary label $y \notin \hat{\mathbf{y}}$)". The loss function is computed,

$$\mathcal{L}(f(y;t,\mathbf{x}),y) = \begin{cases} -\log(f(y;t,\mathbf{x})), & y \in \hat{\mathbf{y}} \\ -\log(1-f(y;t,\mathbf{x})), & y \notin \hat{\mathbf{y}} \end{cases} \quad (4)$$

Finally, we optimize the empirical risk by integrating confidence $g(y;\hat{\mathbf{y}},t,\mathbf{x})$ with the loss function (Equation 1). We will introduce the confidence $g(y;\hat{\mathbf{y}},t,\mathbf{x})$ in detail below.

2.4 Confidence Estimator

The confidence estimator is used to learn the confidence scores $g(y;\hat{\mathbf{y}},t,\mathbf{x})$, which represents the confidence of label y given the token t, text sequence $\mathbf{x}$, and partial label $\hat{\mathbf{y}}$.

$$g(y;\hat{\mathbf{y}},t,\mathbf{x}) = \alpha * c^{A}_{y;t,\mathbf{x}} + (1-\alpha) * c^{M}_{y;t,\mathbf{x}} \quad (5)$$

where the confidence score $c^{M}_{y;t,\mathbf{x}}$ is learned by model and $c^{A}_{y;t,\mathbf{x}}$ is given by annotators. α is a hyper-parameter used to balance these two terms. The annotators will affect the quality of the datasets and we can calculate the prior confidence based on the labeled times of each class. However, prior confidence is biased since the annotators we selected have biases. To address this problem, we also let the model learn the posterior confidence to reduce the biases in prior confidence.

Posterior Confidence We update posterior confidence $c^{M}_{y;t,\mathbf{x}}$ based on true posterior distribution $P(Y=y|T=t,X=\mathbf{x})$ estimated by true posterior estimator $f(y;t,\mathbf{x})$.

$$c^{M}_{y;t,\mathbf{x}} = \begin{cases} \frac{exp(P(Y=y|T=t,X=\mathbf{x}))}{\sum_{\hat{y}\in\hat{\mathbf{y}}} exp(P(Y=\hat{y}|T=t,X=\mathbf{x}))}, & y \in \hat{\mathbf{y}} \\ \frac{exp(P(Y=y|T=t,X=\mathbf{x}))}{\sum_{\hat{y}\notin\hat{\mathbf{y}}} exp(P(Y=\hat{y}|T=t,X=\mathbf{x}))}, & y \notin \hat{\mathbf{y}} \end{cases} \quad (6)$$

We calculate the confidence score for positive and negative labels independently.

Prior Confidence We translate the labeled times $\mathbf{a}$ obtained from annotation into prior confidence $c^{A}_{y;t,\mathbf{x}}$.

$$c^{A}_{y;t,\mathbf{x}} = \begin{cases} \frac{exp(a)}{\sum_{\tilde{a}\in\mathbf{a}} exp(\tilde{a})}, & y \in \hat{\mathbf{y}} \\ 0, & y \notin \hat{\mathbf{y}} \end{cases} \quad (7)$$

Note that both $c^{M}_{y;t,\mathbf{x}}$ and $c^{A}_{y;t,\mathbf{x}}$ are token and content dependent. The annotations are always affected by both the token self and the content of the token. Thus, we model the confidence by considering both the token and content. Finally, we compute the final confidence score $g(y;\hat{\mathbf{y}},t,\mathbf{x})$ via Equation 5, which considers both biases from annotators and models.

We update the parameters θ and confidence score in the M step and E step of the EM algorithm. Specifically, we perform the true posterior estimator and confidence estimator iteratively. The initialization of $c^{M}_{y;t,\mathbf{x}}$ is $\frac{1}{|\hat{\mathbf{y}}|}$ for $y \in \hat{\mathbf{y}}$ and $\frac{1}{|\mathcal{Y}|-|\hat{\mathbf{y}}|}$ for $y \notin \hat{\mathbf{y}}$.

	#Sample	#TIME	#PLACE	#PERSON
Training	1000	6934	958	3518
Dev	440	955	147	351
Test	441	1015	171	356

Table 2: The statistical information of real-world dataset. #Sample means the number of samples in the corresponding dataset. #TIME, #PLACE and #PERSON represent the number of time, place, and person entities.

3 Experimental Setups

In this section, we first introduce one real-world and four synthetic datasets we adopted to evaluate the performance (Section 3.1). Then, we list the selected popular baselines to investigate the validity of our CPLL model (Section 3.2). Finally, we present the implementation details and metrics to replicate the experiment easily (Section 3.3).

3.1 Datasets

Real-World Dataset. To build the real-world dataset, we ask the annotators to label the person, place, and time in the text independently. Each sample is assigned to three annotators with guidelines and several examples. To be specific, we ask three students to label 1000 samples as the training set. The average Kappa value among the annotators is 0.215, indicating that the crowd annotators have low agreement on identifying entities in this data. In order to evaluate the system performances, we create a set of the corpus with gold annotations. Concretely, we randomly select 881 sentences from the raw dataset and let two experts generate the gold annotations. Among them, we use 440 sentences as the development set and the remaining 441 as the test set. Table 2 shows the statistical information of this dataset.

Synthetic Datasets. Inspired by (Rodrigues et al., 2014), we build synthetic datasets by adding

	#Original	r	#Error		
			BI	C	Percent
Weibo	4951	5%	35	134	3.4%
		10%	143	546	13.9%
		20%	494	1706	44.4%
		25%	615	2411	61.0%
Resume	79014	5%	244	2011	2.8%
		10%	920	7361	10.4%
		20%	2979	25408	35.9%
		25%	4145	37585	52.8%
Ontonotes	41203	5%	295	1246	3.7%
		10%	978	4368	12.9%
		20%	3151	14849	43.6%
		25%	4420	20542	60.5%
MSRA	241809	5%	1439	6869	3.4%
		10%	5115	26343	13.0%
		20%	16729	86549	42.0%
		25%	23163	120707	59.4%

Table 3: The statistical information of synthetic datasets. #Original means the number of the tokens labeled as an entity (not O) in the original dataset. BI/C means the number of tokens that have a wrong BI/Category label but the right Category/BI label. Percent = (BI+C)/#Original.

noise on four typical NER datasets: MSRA (Levow, 2006), Weibo (Peng and Dredze, 2015), Ontonotes 4.0 (Weischedel et al., 2011) and Resume (Zhang and Yang, 2018). To simulate a real noise situation, we add noise to the original datasets using four rules: 1) BE (Bound Error) that adds or deletes some tokens of the entity to destroy the bound (change "room 1003" to "(room 1003"); 2) ME (Missing Error) that removes the entity from the label ("David" is not labeled); 3) CE (Category Error) that changes the category of the entity (change "Location" to "Organization"); 4) SE (Segmentation Error) that splits the entity into two entities (change "tomorrow at 10:00 am" to "tomorrow" and "at 10:00 am"). We run each rule randomly with a perturbation rate r, which is set as 10% in the experiments. Additionally, we explore the influence of annotation inconsistency with different rates. Table 3 shows statistical information of these datasets based on token-level majority voting. We can find that a large number of entities are perturbed by our rules. For example, more than 40% tokens labeled as entities are perturbed with a perturbation rate r of 20%.

3.2 Baselines

To verify the effectiveness of our CPLL model, we compare it with several strong and typical baselines, which can be categorized into three groups: voting-based models, partial label learning-based models, and annotator-based models.

- **Voting-based models.** We select two voting-based models, entity-level and token-level voting models. The entity-level voting model obtains the ground truth by voting at the entity level. The token-level voting model calculates the ground truth by voting at the token level. A BERT-based sequence labeling model (Kenton and Toutanova, 2019) is trained based on the ground truth calculated by voting.

- **Partial label learning-based models.** We adopt two classic PLL baselines to utilize the crowd-annotated data with multiple candidate labels. PRODEN-mlp (Lv et al., 2020) adopts a classifier-consistent risk estimator with a progressive identification method for PLL. Wen et al. (2021) propose a Leveraged Weighted (LW) loss for PLL to take the partial and non-partial labels into account, which is proved to be risk consistency. It achieved state-of-the-art results on various computer version tasks. We implement the models by translating the official codes to our NER task.

- **Annotator-based models.** After seeing researchers achieve great success in fully-supervised learning, we are easily going to think about how to gain fully-supervised data from crowd-annotated data when we use crowdsourcing. Seqcrowd (Nguyen et al., 2017) uses a crowd component, a Hidden Markov Model (HMM) learned by the Expectation-Maximization algorithm, to transform crowd-annotated data into fully-supervised data instead of simply voting at token-level or entity-level. When we get the ground truth calculated by this crowd component, we can adopt some efficient fully-supervised learning method to finish the corresponding task.

3.3 Implementation Details and Metrics

We adopt a PyTorch (Paszke et al., 2019) framework Transformers to implement our model based on GPU GTX TITAN X. Chinese-roberta-wwm-ext model (Cui et al., 2019) [1] is used for our true posterior estimator. We utilize Adam optimizer (Kingma and Ba, 2014) to update our model and set different learning rates for the BERT module (0.00002) and the rest module (0.002). The max sequence length

[1] https://huggingface.co/hfl/chinese-roberta-wwm-ext/tree/main

		Real-World		Ontonotes		Weibo		Resume		MSRA
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Test
Ours	CPLL	**90.37**	**90.60**	**79.39**	**81.47**	**69.72**	**68.23**	**96.57**	**96.07**	**95.42**
Voting	Token-level	89.45	90.40	78.17	80.12	67.79	63.81	95.81	95.39	94.68
	Entity-level	89.79	90.04	78.02	79.30	65.59	59.34	95.64	94.88	94.78
PLL	PRODEN-mlp	87.39	87.90	73.04	75.36	66.37	61.85	93.90	94.90	92.46
	LW loss	88.80	89.83	79.07	80.45	69.63	64.26	96.37	95.64	95.35
Annotator	Seqcrowd	-	-	62.80	65.34	47.56	41.49	92.73	93.30	91.90
Upper Bound	Clean data	-	-	79.74	81.47	70.83	68.87	96.64	96.31	95.53

Table 4: The performance of our model and baselines in terms of F1. For real-world dataset, we do not report the results on clean data and Seqcrowd since we do not have ground truth for the training set.

	Real-World		Ontonotes		Weibo		Resume		MSRA
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Test
CPLL	**90.37**	**90.60**	**79.39**	**81.47**	**69.72**	**68.23**	**96.57**	**96.07**	**95.42**
w/o Posterior Confidence	89.51	90.08	79.11	80.42	68.83	65.84	95.74	95.38	94.79
w/o Prior Confidence	90.60	90.94	79.68	80.87	70.57	64.90	96.21	95.70	95.20
w/o Both	86.73	86.32	78.66	80.22	67.33	61.59	95.72	95.23	94.61

Table 5: The performance of ablation studies.

is 512, the batch size is 8 and the dropout rate is 0.1. We search the best α from 0.1 to 0.9 with step 0.1 using the development set. All the baselines use the same settings hyper-parameters mentioned in their paper. Our source code will be available soon after this paper is accepted.

To measure the performance of the models, we adopt Macro-F1 as the metric, which is widely used for NER (Yadav and Bethard, 2018). In particular, we evaluate the performance on the span level, where the answer will be considered correct only when the entire span is matched.

4 Experimental Results

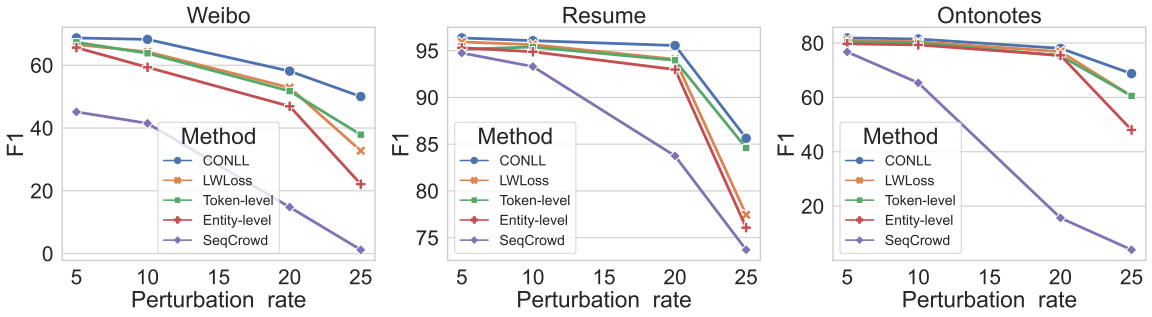
In this section, we conduct a series of experiments to investigate the effectiveness of the proposed CPLL model. Specifically, we compare our model with three kinds of strong baselines (Section 4.1) and do ablation studies to explore the influence of the key parts contained in CPLL (Section 4.2). Also, we investigate the influence of annotation inconsistency (Section 4.3) and hyper-parameter α, which controls the balance of posterior confidence and prior confidence (Section 4.4).

4.1 Main Results

To evaluate the performance of our model, we present the results of compared baselines and our CPLL model (See Table 4). **First**, we can find that our model outperforms all the baselines on both the real-world and synthetic datasets. The labels obtained by voting-based methods (e.g., Token-level voting and entity-level voting) always contain much noise because of the large labeling space and the complexity of this task. For PLL-based models (e.g., PRODEN-mlp and LW loss), they ignore the labeled times by the annotators. Furthermore, annotator-based methods (e.g., Seqcrowd) aim to find the trustworthy label or annotator. Note that Seqcrow does not work on Weibo and performs poorly on Ontonotes. It is because Seqcrow cannot solve the case of small sizes or large noise of datasets, which is also verified in Section 2. All these methods cause information loss which affects the performance of the models largely. Our CPLL model makes use of the crowd-annotated data by translating this task into a PLL task to integrate confidence. **Second**, our CPLL model can reduce the influence of noise effectively. From the results, we observe that CPLL obtains comparable results with the model trained on the clean data. Our confidence estimator can learn the bias generated by annotations effectively via the posterior and prior confidence.

4.2 Ablation Studies

To evaluate the effectiveness of each part contained in our model, we do ablation studies (See Table 5). We remove posterior confidence (w/o Posterior Confidence), prior confidence (w/o Prior Confi-



(a) Weibo (b) Resume (c) Ontonotes

Figure 2: The influence of annotation inconsistency.

dence), and both of them (w/o Both) from CPLL model. For w/o Both, we remove the confidence estimator by setting the confidences as $1/|\hat{\mathbf{y}}|$ for partial labels and 0 for non-partial labels.

From the results, we find the following observations. 1) Confidence estimator can learn the annotation bias effectively. Removing it (w/o Both) reduces more than 4 points in terms of F1 on the test sets over real-world and Weibo datasets. 2) Both posterior confidence and prior confidence are useful for this task. Obviously, prior confidence is vital to leverage the labeled confidence given by annotators. However, prior confidence may exist bias since the annotators are limited. Thus, the posterior confidence learned by the model is also crucial for partial label learning to rectify the prediction.

4.3 Influence of Annotation Inconsistency

We also explore the influence of annotation inconsistency on synthetic datasets with various perturbation rates. Annotation inconsistency is used to model the label quality of crowd-sourcing. The bigger the perturbation rate, the worse the quality of the annotation. We report the results with a rate from 5% to 25% with step 5% over Weibo, Resume ,and Ontonotes datasets (Figure 2).

First, our CPLL model outperforms all the baselines with different perturbation rates. Moreover, the higher the annotation inconsistency, the more our model improves relative to the baselines. Our model can reduce the influence of annotation inconsistency more effectively. Second, several baselines almost do not work with a large perturbation rate (e.g., 25%), while our model can handle it effectively. The F1 score of Seqcrowd is only less than 20 when the rate r is larger than 20%. Third, it is obvious that the annotation quality will affect the performance of the model largely. The higher the inconsistency, the worse the quality of the annotation and the worse the performance of the model.

4.4 Influence of Hyper-parameter α

We further investigate the influence of the hyper-parameter α (in Equation 5), which is used to balance the posterior and prior confidence (Figure 3). The prior confidence demonstrates the labeled confidence given by the annotators, which is biased due to the selection of annotators. To reduce this bias, we enhance our model to estimate the posterior confidence that is learned by the model.

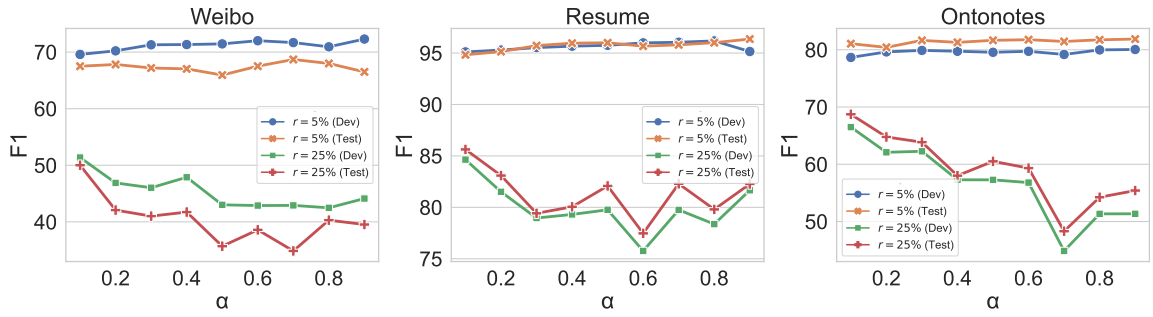
From the figures, we can observe the following observations. First, when the noise is high, the smaller the α, the better the performance. Intuitively, the confidence given by annotators is not reliable when the perturbation rate r is large. Second, when the noise is low, the trend that the larger the α, the better the performance is relatively not as obvious. The reason is that the model can disambiguate the ground truth from the candidates easily since the data is clear. Most of the labels are correct and confidence is not important at this time. All the findings indicate that our confidence estimator can make use of prior confidence and learn posterior confidence effectively.

5 Related Work

In this section, we mainly review the most related works about named entity recognition (Section 5.1) and partial label learning (Section 5.2).

5.1 Named Entity Recognition

Named Entity Recognition (NER) is a research hotspot since it can be applied to many downstream Natural language Processing (NLP) tasks. A well-trained NER model takes language sequence as input and marks out all the entities in the sequence with the correct entity type. NER is widely treated as a sequence labeling problem, a token-level tagging task (Chiu and Nichols, 2015; Akbik et al., 2018; Yan et al., 2019). Also, some of the re-



(a) Weibo (b) Resume (c) Ontonotes

Figure 3: The influence of hyper-parameter α, which is leveraged to control the balance between the posterior and prior confidence.

searchers regard NER as a span-level classification task (Xue et al., 2020; Fu et al., 2021; Alemi et al., 2023). In these works, NER is a fully-supervised learning task based on large-scale labeled data, where each token is asserted with a golden label.

Crowdsourcing platforms (e.g., Amazon Mechanical Turk) are a popular way to obtain large labeled data. Due to the large label space and complexity of NER, the quality of labeled data is low. The ground truth obtained by simple majority voting contains a lot of noise, which limits the performance of the model largely. There is some literature that trains the model from multiple annotators directly (Simpson and Gurevych, 2019; Nguyen et al., 2017). They mainly focus on modeling the differences among annotators to find a trustworthy annotator. In fact, a sentence may not be correctly labeled by all the annotators while they all may label part of the right entities. To address this problem, we translate this task into a partial label learning problem with a prior confidence score.

5.2 Partial Label Learning

Unlike fully-supervised learning, which uses data with golden label $\mathbf{y}$, Partial Label Learning (PLL) asserts a candidate set $\mathcal{Y}$ for each input $\mathbf{x}$ (Zhang et al., 2016; Wang et al., 2023; Lv et al., 2020). Despite the fact that we can not ensure golden label $\mathbf{y}$ always in the candidate set $\mathcal{Y}$, most PLL researchers assume one of the candidate labels is the golden label for simplicity. The existing studies about PLL can be categorized into two groups, average-based methods (Zhang and Yu, 2015) and identification-based methods (Jin and Ghahramani, 2002; Lyu et al., 2019). Average-based methods (Zhang and Yu, 2015; Hüllermeier and Beringer, 2006) intuitively treat the candidate labels with equal importance. The main weakness of these algorithms is that the false positive may severely distract the model with wrong label information. Recently, identification-based methods (Jin and Ghahramani, 2002; Wang et al., 2023) are proposed to identify the truth label from the candidates by regarding the ground truth as a latent variable. More and more literature pays attention to representative methods (Lyu et al., 2019; Nguyen and Caruana, 2008), self-training methods (Wen et al., 2021), loss function adjustments (Wu and Zhang, 2018).

However, most of the current work focuses on image classification or text classification tasks, while how to model the confidence for NER is not well studied. The sequence labeling task aims to identify the entities in the sentence with an entity type in the token level. Thus, how to model the token self and its content also plays an important role in this task. To address this problem, we design a confidence estimator to predict the token- and content-dependent confidence based on the prior confidence given by annotators.

6 Conclusion and Future Work

In this paper, we translate crowd-annotated NER into a PLL problem and propose a `CPLL` model based on an EM algorithm. To rectify the model's prediction, we design a confidence estimator to predict token- and content-dependent confidence by incorporating prior confidence with posterior confidence. We conduct the experiments on one real-world dataset and four synthetic datasets to evaluate the performance of our proposed `CPLL` model by comparing it with several state-of-the-art baselines. Moreover, we do ablation studies to verify the effectiveness of the key components and explore the influence of annotation inconsistency.

In the future, we would like to investigate the performance of our model on other sequence labeling tasks.

Limitations

Although our work shows that our CPLL model can learn from crowd-annotated NER data well, there are at least two limitations. First, we set the hyperparameter α manually. It would be better if we could design a strategy to learn a $alpha$ adaptive value for each sample atomically. Second, though we mainly experiment on NER tasks, our model can be applied to all sequence labeling tasks, such as part-of-speech tagging (POS), Chinese word segmentation, and so on. We would like to explore it in further work.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No.62206057), Shanghai Rising-Star Program (23QA1400200), Natural Science Foundation of Shanghai (23ZR1403500), and CCF-Tencent Open Fund.

References

Alan Akbik, Duncan A. J. Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. *international conference on computational linguistics*.

Alexander Alemi, Ian Fischer, Joshua Dillon, Jacob Devlin, Ming-Wei Chang, Kenton Lee, Marco Federici, Anjan Dutta, Patrick Forré, Nate Kush, Robert Geirhos, Jörn-Henrik Jacobsen, Richard Michaelis, and Wieland Zemel. 2023. Miner: Improving out-of-vocabulary named entity recognition from an information theoretic perspective. *meeting of the association for computational linguistics*.

Nguyen Bach and Sameer Badaskar. 2007. A review of relation extraction. *Literature review for Language and Statistics II*, 2:1–15.

Jason P.C. Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*.

Timothee Cour, Ben Sapp, and Ben Taskar. 2011. Learning from partial labels. *The Journal of Machine Learning Research*, 12:1501–1536.

Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-training with whole word masking for chinese bert. *arXiv: Computation and Language*.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Lei Feng and Bo An. 2019. Partial label learning with self-guided retraining. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3542–3549.

Lei Feng, Jiaqi Lv, Bo Han, Miao Xu, Gang Niu, Xin Geng, Bo An, and Masashi Sugiyama. 2020. Provably consistent partial-label learning. *Advances in Neural Information Processing Systems*, 33:10948–10960.

Jinlan Fu, Xuanjing Huang, and Pengfei Liu. 2021. Spanner: Named entity re-/recognition as span prediction. *meeting of the association for computational linguistics*.

Eyke Hüllermeier and Jürgen Beringer. 2006. Learning from ambiguously labeled examples. *Intelligent Data Analysis*, 10(5):419–439.

Rong Jin and Zoubin Ghahramani. 2002. Learning with multiple labels. *Advances in neural information processing systems*, 15.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. 2019. Nlnl: Negative learning for noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 101–110.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117.

Jiaqi Lv, Miao Xu, Lei Feng, Gang Niu, Xin Geng, and Masashi Sugiyama. 2020. Progressive identification of true labels for partial-label learning. In *International Conference on Machine Learning*, pages 6500–6510. PMLR.

Gengyu Lyu, Songhe Feng, Tao Wang, Congyan Lang, and Yidong Li. 2019. Gm-pll: graph matching based partial label learning. *IEEE Transactions on Knowledge and Data Engineering*, 33(2):521–535.

An T Nguyen, Byron C Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and predicting sequence labels from crowd annotations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2017, page 299. NIH Public Access.

Nam Nguyen and Rich Caruana. 2008. Classification with partial labels. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–559.

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

Nanyun Peng and Mark Dredze. 2015. Named entity recognition for chinese social media with jointly trained embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 548–554.

Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Sequence labeling with multiple annotators. *Machine learning*, 95(2):165–181.

Edwin Simpson and Iryna Gurevych. 2019. A Bayesian approach for sequence tagging with crowds. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1093–1104, Hong Kong, China. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789.

Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. 2023. Pico: Contrastive label disambiguation for partial label learning. *Learning*.

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Hongwei Wen, Jingyi Cui, Hanyuan Hang, Jiabin Liu, Yisen Wang, and Zhouchen Lin. 2021. Leveraged weighted loss for partial label learning. In *International Conference on Machine Learning*, pages 11091–11100. PMLR.

Xuan Wu and Min-Ling Zhang. 2018. Towards enabling binary decomposition for partial label learning. In *IJCAI*, pages 2868–2874.

Mengge Xue, Bowen Yu, Zhenyu Zhang, Tingwen Liu, Yue Zhang, and Bin Wang. 2020. Coarse-to-fine pre-training for named entity recognition. *empirical methods in natural language processing*.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *27th International Conference on Computational Linguistics, COLING 2018*, pages 2145–2158. Association for Computational Linguistics (ACL).

Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: Adapting transformer encoder for named entity recognition. *arXiv: Computation and Language*.

Yan Yan and Yuhong Guo. 2020. Partial label learning with batch label correction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6575–6582.

YaoSheng Yang, Meishan Zhang, Wenliang Chen, Wei Zhang, Haofen Wang, and Min Zhang. 2018. Adversarial learning for chinese ner from crowd annotations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Min-Ling Zhang and Fei Yu. 2015. Solving the partial label learning problem: An instance-based approach. In *Twenty-fourth international joint conference on artificial intelligence*.

Min-Ling Zhang, Bin-Bin Zhou, and Xu-Ying Liu. 2016. Partial label learning via feature-aware disambiguation. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1335–1344.

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564.

Jie Zhou, Qi Zhang, Qin Chen, Liang He, and Xuan-Jing Huang. 2022. A multi-format transfer learning model for event argument extraction via variational information bottleneck. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1990–2000.

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?
Our work does not have any potential risks.

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

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

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ 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)?
No response.

☐ 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?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ 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.
No response.

C ☑ Did you run computational experiments?

3.3 Implementation Details and Metrics

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
3.3 Implementation Details and Metrics

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?
3.3 Implementation Details and Metrics

☒ 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?
We run our model using the same seed and select the best based on the development set.

☐ 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.)?
Not applicable. Left blank.

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

3.1 Datasets

☑ 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.?
3.1 Datasets

☑ 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)?
3.1 Datasets

☑ 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?
3.1 Datasets

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.