DynClean: Training Dynamics-based Label Cleaning for Distantly-Supervised Named Entity Recognition

Anonymous ACL submission

Abstract

Distantly Supervised Named Entity Recognition (DS-NER) has attracted attention due to its scalability and ability to automatically generate labeled data. However, distant annotation introduces many mislabeled instances, limiting its performance. Most of the existing work attempt to solve this problem by developing intricate models to learn from the noisy labels. An alternative approach is to attempt to clean the labeled data, thus increasing the quality of distant labels. This approach has received little attention for NER. In this paper, we propose a training dynamics-based label cleaning approach, which leverages the behavior of a model as training progresses to characterize the distantly annotated samples. We also introduce an automatic threshold estimation strategy to locate the errors in distant labels. Extensive experimental results demonstrate that: ❶ models trained on our cleaned DS-NER datasets, which were refined by directly removing identified erroneous annotations, achieve significant improvements in F1-score, ranging from 3.19% to 8.95%; and ❷ our method outperforms numerous advanced DS-NER approaches across four datasets.

1 Introduction

Named entity recognition (NER), which aims to identify spans in text that refer to pre-defined entity types such as person, location, or organization names, has wide downstream applications such as question answering (Lan et al., 2021), information retrieval (Choudhary et al., 2023), and knowledge graph construction (Ji et al., 2022). However, NER performance often relies on human annotated high-quality training data, which is time-consuming and expensive, making the development of robust NER models for real-world applications hard. Distantly Supervised Named Entity Recognition (DS-NER) (Ren et al., 2015; Fries et al., 2017; Shang et al., 2018; Liang et al., 2020; Xu et al., 2023; Pan et al., 2023; Wu et al., 2023) has been proposed to automatically generate labeled data for training NER models. In general, DS-NER leverages existing resources such as knowledge bases and rule-based string matching methods like regular expression to automatically produce the training data, which is generally referred as *weakly annotated data* (Zhang et al., 2021a). Since unlabeled data is easy to collect, DS-NER can significantly reduce the annotation efforts and obtain large-scale datasets. The weakly annotated data, however, suffers from the introduction of noisy labels. There are two typical issues in DS-NER: (1) False Negative: when an entity is not detected due to the limited coverage of knowledge bases and (2) False Positive: when annotated spans of text is incorrectly detected as entities or assign incorrect entity types. For example, in Figure 1 the organization entity "Washington" was incorrectly annotated as a person type, since the rule-based string matching cannot distinguish between two entity types (person and organization in this case) with the same surface form; "Tamil" is not recognized as an entity because it is not included in the knowledge base used.

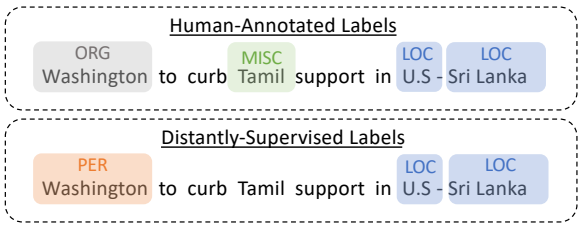

Figure 1: A typical distantly-supervised annotation can be subject to two types of error: (1) False Positive: An entity is recognized to incorrectly type, e.g., "Washington" and (2) False Negative: An entity is recognized as non-entity, e.g.,"Tamil".

Models trained on such DS-NER dataset can easily overfit to the noisy labels, leading to poor performance (Tänzer et al., 2022). Many approaches have been proposed to alleviate these limitations. Negative sampling (Li et al., 2021, 2022; Xu et al.,

2023) and positive unlabeled (PU) learning (Peng et al., 2019; Zhou et al., 2022; Wu et al., 2023) have been proposed to handle the false negative annotation issues in DS-NER. But these works overlook false positives, which limit their performance. Another line of work employs self-training (Meng et al., 2021; Zhang et al., 2021b; Liang et al., 2020) by utilizing soft labels generated by a teacher model. They require a high-quality teacher model to produce reliable soft labels for the student model, often resulting in the inclusion of additional modules. A common challenge with self-training methods is their complex architectures, which require the training of multiple models across several iterations. Despite those efforts, the research on DS-NER has yet to thoroughly investigate the improvement of the quality of distantly annotated labels.

In this paper, we propose the DynClean, a training dynamics-based label cleaning approach for DS-NER. Unlike most existing methods that focus on learning from noisy labels, *our proposed approach aims to improve the quality of data generated by distant supervision annotation*. In DynClean, we leverage the training dynamics (i.e., the behavior of a model as training progresses) to reveal the characteristics of each data sample in DS-NER datasets. To differentiate between clean and mislabeled samples using the obtained data characteristics, we introduce an automatic threshold estimation strategy. We show that DynClean can effectively locate two types of errors: false positives and negatives. Finally, we filter out the identified mislabeled distant samples and then obtain "mostly clean"data for training. We validated our proposed approach across different NER models on four popular DS-NER datasets and achieved a consistent performance boost, with improvements in the F1 score ranging from 3.18% to 8.95%. These results empirically demonstrate that enhancing the quality of distantly annotated data improve the performance of DS-NER tasks. Furthermore, models trained on the four DS-NER datasets processed by DynClean outperform current state-of-the-art methods, achieving up to 4.53% F1 score improvements despite using fewer samples in training.

2 Related Work

Distantly-Supervised NER. Several Negative sampling (Li et al., 2021, 2022; Xu et al., 2023) methods have been proposed to reduce the negative impact of unlabeled entities in training data. Other works leverage the positive unlabeled (PU) learning and include potentially noisy positive samples along with unlabeled data (Peng et al., 2019; Zhou et al., 2022). Another line of works is based on the self-training strategy (Jie et al., 2019; Liang et al., 2020; Zhang et al., 2021c; Ma et al., 2023; Meng et al., 2021). In general, they design a teacher-student network to iteratively refine the entity labels and reduce both false positive and false negative samples. (Ying et al., 2022) proposed CReDEL to leverage contrastive learning to learn the refinement knowledge of distant annotation. (Wu et al., 2023) design the MProto to model the token-prototype assignment problem as an optimal transport problem to handle intra-class variance issues. In our work, we introduce a training dynamics-based method that denoises DS-NER datasets automatically without requiring additional components.

As discussed in (Zhou et al., 2022; Xu et al., 2023), self-training based methods are post-processing steps . Thus, self-training (Liang et al., 2020; Zhang et al., 2021c; Meng et al., 2021; Ma et al., 2023; Si et al., 2023) approaches are orthogonal to data cleaning approaches (like ours), which are pre-processing steps; we do not discuss them further in this paper.

Training Dynamics. Training dynamics are defined as statistics calculated of the model behavior (e.g., the output logit values) on each sample across the training process. It can be used to evaluate the characteristics and quality of each individual sample within a dataset. The frequently used metrics include Confidence and Variability proposed by (Swayamdipta et al., 2020), and the Area Under Margin (AUM) proposed by (Pleiss et al., 2020). For each sample, Confidence and Variability are the mean and standard deviation of the gold label probabilities over the training epochs, respectively. AUM finds the difference in logit value of the assigned class (gold label) and the highest logit value among the non-assigned classes averaged over training epochs for every sample. These metrics capture the data properties and can be used to evaluate the data quality. Most works obtained the data characteristics from the training dynamics and combined it with different techniques, such as data augmentation (Park and Caragea, 2022) and curriculum learning (Sar-Shalom and Schwartz, 2023). In our work, we adapt the training dynamics metrics to diagnose the DS-NER datasets and identify the noisy instances.

3 Method

We introduce our proposed method, DynClean. DynClean leverages a model's behavior on individual samples in the DS-NER across training epochs (i.e., the training dynamics) to create characteristics of data samples. Then, using the proposed automatic threshold estimation strategy, we distinguish clean and mislabeled data within the DS-NER dataset. In this section, we first give a brief overview of the span-based NER model, followed by a detailed presentation of the proposed DynClean.

3.1 Background

Span-based NER Model. Following previous works (Li et al., 2022; Si et al., 2023), we employ the span-based NER architecture. Given a sentence with n tokens, $[t_1, t_2, ..., t_n]$, we take a pre-trained language model as the encoder to get the contextualized representation:

$$[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n] = \text{Encoder}([t_1, t_2, ..., t_n]) \quad (1)$$

where $\mathbf{h}_i$ is the representation for token t_i. We derive the representation for one span sample x which starts at position i and ends at position j as:

$$\mathbf{x} = \mathbf{h}_i \oplus \mathbf{h}_j \oplus \mathbf{D}(j - i) \quad (2)$$

where $\oplus$ is the concatenation operation, $\mathbf{D}(j - i)$ is a learnable embedding to encode the span length feature, and the span length is limited to $0 \leq j - i \leq L$. A feed-forward neural network (FFNN) is used to obtain the logits $\mathbf{z}$, and softmax computes the probability distribution of label y:

$$\mathbf{z} = \text{FFNN}(\mathbf{x}) \quad (3)$$

$$P(y|\mathbf{x}) = \text{softmax}(\mathbf{z}) \quad (4)$$

Training. Cross-entropy loss function is calculated on all spans during training:

$$\mathcal{L} = -\sum_{\mathbf{x} \in \mathbf{X}} \log P(y^*|\mathbf{x}) \quad (5)$$

where y^* represents the target label.

Negative Sampling. Previous studies (Li et al., 2022, 2021; Xu et al., 2023) have demonstrated that negative sampling can effectively mitigate the impact of false negatives in DS-NER. Motivated by the (Xu et al., 2023), we consider only utilizing the top-N_r negative samples (TopNeg), which exhibit high similarities with all positive samples, for training the DS-NER model. TopNeg reduces the number of false negative samples involved in training, thereby enhancing DS-NER performance. Specifically, for each batch of training data, we obtain the span sets of positive samples $\mathbf{X}^{\text{pos}} = \{..., \mathbf{x}^{\text{pos}}, ...\}$, and negative samples $\mathbf{X}^{\text{neg}} = \{..., \mathbf{x}^{\text{neg}}, ...\}$, where $\mathbf{X} = \mathbf{X}^{\text{pos}} \cup \mathbf{X}^{\text{neg}}$. Subsequently, for each negative sample $\mathbf{x}^{\text{neg}} \in \mathbf{X}^{\text{neg}}$, we compute the similarity score ϕ:

$$\phi(\mathbf{x}^{\text{neg}}, \mathbf{X}^{\text{pos}}) = \frac{1}{M} \sum_{\mathbf{x}^{\text{pos}} \in \mathbf{X}^{\text{pos}}} \frac{\mathbf{x}^{\text{neg}}}{\|\mathbf{x}^{\text{neg}}\|} \cdot \frac{\mathbf{x}^{\text{pos}}}{\|\mathbf{x}^{\text{pos}}\|} \quad (6)$$

where M denotes the number of positive samples in one batch of data.

When training the model with TopNeg, the negative samples in each batch are ranked according to their similarity scores, as defined in Equation 6. Subsequently, the top-N_r negative samples, along with all positive samples, are selected for loss calculation. The Equation 5 is modified as:

$$\begin{aligned} \mathcal{L} = &-\sum_{\mathbf{x}^{\text{pos}} \in \mathbf{X}^{\text{pos}}} \log P(y^*|\mathbf{x}^{\text{pos}}) \\ &-\sum_{\tilde{\mathbf{x}}^{\text{neg}} \in \tilde{\mathbf{X}}^{\text{neg}}} \log P(y^*|\tilde{\mathbf{x}}^{\text{neg}}) \end{aligned} \quad (7)$$

where $\tilde{\mathbf{X}}^{\text{neg}}$ is the set of the selected negative samples.

3.2 Proposed Approach: DynClean

We detail our proposed method, DynClean, which differentiates clean and mislabeled data based on the characteristics of each sample, derived from the training dynamics. We first define the training dynamics and corresponding metrics, and then use these to reveal the characteristics of each data sample. We then introduce our threshold estimation strategy and procedure of DynClean in cleansing distant labels.

Defining training dynamics. Training dynamics can be defined as any model behavior during the training, such as the area under the margin (AUM) between logit values of the target label and the other largest label (Pleiss et al., 2020) or the mean of predicted probabilities (Confidence) of target label (Swayamdipta et al., 2020). This section focuses on AUM in our primary experiments, with a discussion on Confidence in Section 6.3.

Given a sample $\mathbf{x}$ with assigned label y^* (potentially erroneous), we compute $\text{AUM}(\mathbf{x}, y^*)$ as the area under the margin averaged across all training epochs E. The margin at epoch e is defined as:

$$M^e(\mathbf{x}, y^*) = \mathbf{z}^e_{y^*}(\mathbf{x}) - \max_{k \neq y^*} \mathbf{z}^e_k(\mathbf{x}) \quad (8)$$

where $M^e(\mathbf{x}, y^*)$ is the margin of sample $\mathbf{x}$ with label y^*, $\mathbf{z}^e_{y^*}(\mathbf{x})$ is the logit corresponding to the label y^*, and $\max_{k \neq y^*} \mathbf{z}^e_k(\mathbf{x})$ is the largest non-assigned logit with label k not equal to y^*. The margin measures the difference between a assigned label and the model's predictions at each epoch e. The $\text{AUM}(\mathbf{x}, y^*)$ across E epochs is computed by:

$$\text{AUM}(\mathbf{x}, y^*) = \frac{1}{E} \sum_{e=1}^{E} M^e(\mathbf{x}, y^*) \tag{9}$$

Precisely, AUM considers each logit value and measures how much the assigned label logit value differs from the other largest logit value. A low AUM signifies that the assigned label is likely incorrect, while a larger margin indicates the assigned label is likely correct. Consequently, AUM can be utilized to distinguish mislabeled samples.

Threshold Estimation. A threshold value is required to separate mislabeled from clean samples. However, given the varying noise distribution across different datasets, determining this threshold is a costly hyper-parameter tuning procedure. Therefore, we construct threshold samples to simulate mislabeled data, and train the model to compute the training dynamics metric values for threshold samples. Data with similar or worse metric values than threshold samples can be assumed as mislabeled. We select a subset of both positive and negative samples from the training data and reassign their labels to a non-existent class, thereby constructing threshold samples. Assuming we have N_p positive samples (i.e. entity spans that belong to c entity types). We use stratified sampling (based on the distribution of the original entity types) to randomly select $N_\text{p}/(c+1)$ samples as positive threshold samples. Then we assign a fake label $c+1$ to these positive threshold samples. As for negative threshold samples, we sample the same number $(N_\text{p}/(c+1))$ of negative samples and assign the fake label $c+1$ as well. A model is trained on the constructed datasets with threshold samples to compute the metric values at the end of training. Positive and negative threshold samples are ranked separately, employing the threshold samples at the k_posth for positives and k_negth for negatives percentile to determine the positive threshold τ_{pos} and negative threshold τ_{neg}. Constructing threshold samples for both positives and negatives is crucial for identifying unlabeled negatives and mislabeled positives in the distant annotation process in our DS-NER case.

Algorithm 1 DynClean

Require: Distantly annotated data $\mathcal{D} = (x_i, y_i)_{i=1,\ldots,n}$, model f, hyperparameter k_pos and k_neg;
// Threshold Estimation
$T_\text{pos} \leftarrow$ Stratified sampling $N_\text{p}/(c+1)$ positives
$T_\text{neg} \leftarrow$Random sampling $N_\text{p}/(c+1)$ negatives
Train f for E epochs and compute AUM for each sample in T_pos and T_neg as in Eq. 9
Rank AUM for T_pos and T_neg separately
$\tau_\text{pos} \leftarrow k_\text{pos}$th at T_pos
$\tau_\text{neg} \leftarrow k_\text{neg}$th at T_neg
// Noisy data cleaning
$\mathcal{D}' \leftarrow \varnothing$, re-initialize model f
Train f for E epochs and compute $\text{AUM}(x_i, y_i)$ for each sample i as in Eq. 9
for each $(x_i, y_i) \in \mathcal{D}$ **do**
  if $y_i > 0$ **then** ▷ Positive samples
    if $\text{AUM}(x_i, y_i) \geq \tau_\text{pos}$ **then**
      $\mathcal{D}' \leftarrow \mathcal{D}' \cup (x_i, y_i)$
  else ▷ Negative samples
    if $\text{AUM}(x_i, y_i) \geq \tau_\text{neg}$ **then**
      $\mathcal{D}' \leftarrow \mathcal{D}' \cup (x_i, y_i)$
return $\mathcal{D}'$

Applying DynClean. Given a DS-NER dataset, we need to utilize a model to train on it in order to gather training dynamics metric values. Specifically, we begin with the proposed automatic threshold estimation strategy, which involves constructing threshold samples and collecting their training dynamics to compute the corresponding metric values. Subsequently, we collect the training dynamics metric values for all samples in the original DS dataset. Finally, we separately filter the positive and negative samples based on the estimated threshold. Thus, our method can be applied to various models. The details of our DynClean procedure when using AUM are summarized in Algorithm 1. Our method is also applicable to other training dynamic metrics, as presented in Section 6.3.

4 Experiments

4.1 Datasets

Our approach is evaluated using four DS-NER datasets: CoNLL03 (Tjong Kim Sang and De Meulder, 2003), WikiGold (Balasuriya et al., 2009), WNUT16 (Godin et al., 2015), and BC5CDR (Wei et al., 2016). Originally, these datasets were human-annotated, and subsequently, they were re-annotated via distant supervision as reported in (Liang et al., 2020; Zhou et al., 2022; Shang et al., 2018). The statistics for the four datasets are presented in Table 5 of Appendix A. The distantly supervised data are used for training, while the human-annotated development and test sets are utilized for evaluation.

4.2 Experimental Setup

To validate the effectiveness of our proposed method, we first conduct cleaning experiments on four datasets with our DynClean framework using different base models. Subsequently, we compare the models trained on our cleaned datasets with existing DS-NER studies. We describe below the base models used in our work.

Base Models. We employ span-based NER model mentioned in Section 3.1, integrating different encoders and their TopNeg variants, as our base models. We evaluate BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) as encoders, along with their TopNeg variants, on CoNLL03, Wikigold, and WNUT16. The four base models are denoted by BERT, BERT-TopNeg, RoBERTa, and RoBERTa-TopNeg. For BC5CDR, a domain-specific NER task, we exclusively employ BioBERT (Lee et al., 2020). For ease of presentation, in the context of BC5CDR, BERT and BERT-TopNeg are used to represent its base models.

Baselines. We compare our proposed method with several existing DS-NER works, including: **KB-Matching** employs knowledge bases to annotate the test sets. **AutoNER** (Shang et al., 2018) modifies the standard CRF to handle noisy labels. **BOND** (Liang et al., 2020) proposes an early stopping approach to prevent model from overfitting to noisy labels. **bnPU** (Peng et al., 2019) formulates the DS-NER task as a positive unlabelled learning problem and uses the mean absolute error as the noise-robust objective function. **Span-NS** (Li et al., 2021), **Span-NS-V** (Li et al., 2022) are the negative sampling approaches, which aim to reduce the false negative samples used for training. **CReDEL** (Ying et al., 2022) adopts a distant label refinement model via contrastive learning. **Conf-MPU** (Zhou et al., 2022) is a two-stage approach, with the first stage estimating the confidence score of being an entity and the second stage incorporating the confidence score into the positive unlabelled learning framework. **MProto** (Wu et al., 2023) models the token-prototype assignment problem as an optimal transport problem to handle intra-class variance issue.

Additionally, we provide fully supervised methods for comparison, including RoBERTa and BiLSTM-CRF (Ma and Hovy, 2016). We also conduct zero-shot evaluations with large language models (LLMs), including ChatGPT and LLaMA3.1-70B.

4.3 Implementation Details

We tune the kth of the threshold samples for threshold estimation, with increments of 5%, ranging from 80% to 100%. The k_{neg}th for computing the threshold in negative threshold samples is set at 90% across all datasets. For positive threshold samples, the tuned k_{pos}th is set at 100% for CoNLL03, Wikigold, and WNUT16, and at 90% for BC5CDR. A more detailed discussion on the effectiveness of our threshold estimation strategy can be found in the Appendix B. All reported experimental results represent the average of five runs, each with a different random seed[1]. More experimental details are provided in the Appendix C.

5 Results

In this section, we report the results in two parts: ❶ for each dataset, we train different base models on the original distantly supervised data and on various cleaned versions of the data; and ❷ we compare the base models trained on our best-cleaned data with the baseline methods.

5.1 Cleaning Results

In our cleaning experiments, we apply the four base models in our DynClean framework to clean each dataset. Specifically, we use $\mathcal{D}$ to denote an original distant dataset, and $\mathcal{D}'_1$, $\mathcal{D}'_2$, $\mathcal{D}'_2$, and $\mathcal{D}'_4$ to respectively represent the datasets cleaned using training dynamics from BERT, RoBERTa, BERT-TopNeg, and RoBERTa-TopNeg. We then train the four base models on the original distant data and the different cleaned data.

Table 1 gives the entire set of results. From the results, we found that the model's performance consistently improved when it was trained with cleaned data compared to the performance with the original distant data. Specifically, for CoNLL03, Wikigold, and WNUT16, models trained on $\mathcal{D}'_4$ consistently achieve the best performance. Therefore, we refer $\mathcal{D}'_4$ as the best-cleaned data for these datasets. For BC5CDR, it is $\mathcal{D}'_3$. Particularly, RoBERTa trained by best-cleaned Wikigold ($\mathcal{D}'_4$) shows an 8.67% improvement in F1 score when compared with original distant data. We also have following observations: ❶ For each dataset, the model used in DynClean to obtain the best-cleaned data also has the best performance in the original distant data. RoBERTa-TopNeg, which obtains the best-cleaned data for CoNLL03, Wikigold, and WNUT16, also performs

[1]Code and data will be released once the work is accepted.

Models	Data Ver.	CoNLL03			WikiGold			WNUT16			BC5CDR		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	$\mathcal{D}$	**89.75**	63.08	72.09	56.12	44.37	49.47	58.18	36.22	44.55	**89.44**	67.57	76.98
	$\mathcal{D}'_1$	88.77	65.61	75.44	58.18	46.29	51.56	57.60	38.19	45.93	87.07	71.67	78.62
	$\mathcal{D}'_2$	89.19	67.92	77.11	58.09	47.94	52.53	58.54	37.98	46.02	-	-	-
	$\mathcal{D}'_3$	86.79	77.12	81.66	47.24	**57.91**	52.02	53.25	**41.60**	46.51	80.02	**83.06**	**81.50**
	$\mathcal{D}'_4$	87.29	**80.40**	**83.70**	**60.75**	56.71	**58.65**	**58.75**	40.25	**47.74**	-	-	-
RoBERTa	$\mathcal{D}$	**90.16**	65.88	76.13	60.63	44.65	51.42	**62.72**	43.45	51.34	-	-	-
	$\mathcal{D}'_1$	89.99	67.76	77.31	57.92	48.30	52.65	61.85	43.65	51.18	-	-	-
	$\mathcal{D}'_2$	89.51	70.10	78.61	56.47	51.12	53.66	62.33	43.86	51.44	-	-	-
	$\mathcal{D}'_3$	87.96	81.47	84.59	51.24	**61.90**	56.02	61.34	47.44	53.50	-	-	-
	$\mathcal{D}'_4$	88.41	**82.77**	**85.50**	**60.94**	59.31	**60.09**	58.22	**51.35**	**54.52**	-	-	-
BERT-TopNeg	$\mathcal{D}$	79.42	79.77	79.58	52.99	47.83	50.09	49.05	**42.96**	45.80	79.22	81.35	80.27
	$\mathcal{D}'_1$	82.09	77.98	79.98	57.59	47.45	51.98	**58.56**	37.92	46.03	**82.79**	76.52	79.53
	$\mathcal{D}'_2$	85.01	77.08	80.84	58.17	47.72	52.43	58.35	38.96	46.71	-	-	-
	$\mathcal{D}'_3$	84.72	80.24	82.41	48.87	55.91	52.14	54.07	41.99	47.21	81.41	**81.48**	**81.42**
	$\mathcal{D}'_4$	**85.71**	**81.16**	**83.37**	**61.85**	**55.91**	**58.72**	55.00	42.05	**47.61**	-	-	-
RoBERTa-TopNeg	$\mathcal{D}$	82.95	78.56	80.70	52.77	54.86	53.80	60.69	45.21	51.56	-	-	-
	$\mathcal{D}'_1$	83.76	80.56	82.13	59.10	50.12	54.23	59.97	44.56	51.01	-	-	-
	$\mathcal{D}'_2$	85.50	77.28	81.18	55.10	54.31	54.69	58.26	50.11	53.83	-	-	-
	$\mathcal{D}'_3$	**86.74**	81.40	83.98	51.15	58.57	54.56	60.20	48.01	53.28	-	-	-
	$\mathcal{D}'_4$	86.34	**83.17**	**84.72**	**60.56**	**59.30**	**59.88**	**58.07**	**51.38**	**54.45**	-	-	-

Table 1: Results of four base models trained on original and different versions of cleaned DS-NER datasets. $\mathcal{D}$ denotes the original distantly annotated data, and $\mathcal{D}'_1$, $\mathcal{D}'_2$, $\mathcal{D}'_3$, and $\mathcal{D}'_4$ to respectively represent the datasets cleaned using training dynamics from BERT, RoBERTa, BERT-TopNeg, and RoBERTa-TopNeg. A bold score indicates that the model trained on the corresponding data version achieves the best performance.

the best on their original distant data. For BC5CDR, the model is BERT-TopNeg. We believe this is because, when the model used in DynClean has better performance on the original distant data, the training dynamics obtained are more capable to reflect the "true" data sample characteristics for cleaning; ❷ When models are trained on the best-cleaned data, it is not necessary to employ TopNeg. For the best-cleaned CoNLL03 ($\mathcal{D}'_4$), WNUT16 ($\mathcal{D}'_4$), and BC5CDR ($\mathcal{D}'_3$), models trained on them without TopNeg achieve better performances. Only training BERT with TopNeg on the WikiGold ($\mathcal{D}'_4$) exhibits a *slight* improvement. This indicates that our proposed approach removes a large enough amount of false negatives, which obviates the necessity for employing additional negative sampling techniques. Our additional experiments in Appendix D also show the TopNeg is not necessary when training models on the *human-annotated* data as well.

5.2 Baselines Comparison

The comparison against the baselines is summarized in Table 2. As our method, we present the results of BERT and RoBERTa trained on the best-cleaned version of each dataset, i.e., the CoNLL03 ($\mathcal{D}'_4$), Wikigold ($\mathcal{D}'_4$), WNUT16 ($\mathcal{D}'_4$), and BC5CDR ($\mathcal{D}'_3$). We first observe that LLMs still face certain challenges in the NER task, with a significant performance gap compared to fully supervised RoBERTa. This result is similar to previous study (Qin et al., 2023), indicating that LLMs still face significant challenges in NER task. Our approach also outperforms all LLMs across four datasets. We furthre found that LLMs outperform KB-Matching, indicating the feasibility of using LLMs for distant annotation. We have provided a related discussion in Appendix E. Despite using *less* data for model training, we outperform all previous approaches with relatively balanced precision and recall scores. When comparing with Conf-MPU$_{\text{BERT}}$, BERT trained on the cleaned CoNLL03 and BC5CDR achieves an F1 score improvement of 4.54% and 4.28%, respectively. Our method also outperforms Conf-MPU$_{\text{LBiLSTM}}$, which is enhanced with lexicon feature engineering, with the F1 score improvement of 3.68% and 1.43%, respectively. Compared to MProto$_{\text{BERT}}$, we have 4.12% F1 improvement on the CoNLL03 dataset. As for the CReDEL which also focuses on enhancing the data quality of distantly supervised datasets, our approach outperforms it by 2.9% F1 score on the BC5CDR dataset. Our approach also outperforms multiple baselines that utilize negative sampling techniques. The main reason our method achieves significant improvement is that we removed a large number of false negatives and false positives from the distantly annotated data.

6 Analysis and Discussion

In this section, we provide the ablation study and case study to better understand the effectiveness of our proposed method. In addition, we provide experimental results of utilizing another frequently used training dynamics metric in our DynClean.

Methods	CoNLL03			WikiGold			WNUT16			BC5CDR		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Fully Supervised												
RoBERTa	90.05	92.48	91.25	85.33	87.56	86.43	51.76	52.63	52.19	88.41	87.28	89.56
BiLSTM-CRF	89.14	91.10	90.11	55.40	54.30	54.90	60.01	46.16	52.18	89.09	62.22	73.27
LLMs Evaluation												
Llama-3.1-70B	71.08	83.37	76.74	57.67	48.15	52.48	50.00	45.46	47.62	67.23	55.87	61.03
ChatGPT	74.29	84.23	78.95	63.51	48.61	55.07	54.55	50.00	52.17	80.61	68.91	74.30
Distantly Supervised												
KB-Matching	81.13	63.75	71.40	47.90	47.63	47.76	40.34	32.22	35.83	86.39	51.24	64.32
AutoNER‡	60.40	75.21	67.00	52.35	43.54	47.54	18.69	43.26	26.10	77.52	82.63	79.99
bnPU †	82.97	74.38	78.44	-	-	-	-	-	-	77.06	48.12	59.24
$\text{BOND}_{\text{RoBERTa}}$	83.76	68.90	75.61	49.17	54.50	51.55	53.11	41.52	46.61	-	-	-
CReDEL	-	-	-	-	-	-	-	-	-	71.70	**86.80**	78.60
Span-NS ‡	80.41	71.35	75.61	51.05	48.27	49.62	53.51	39.76	45.62	**86.90**	73.49	79.64
Span-NS-V ‡	80.19	72.91	76.38	50.91	48.43	49.64	47.78	44.37	46.01	86.67	73.52	79.56
BERT-TopNeg‡	82.72	77.71	80.08	55.47	48.57	50.65	55.28	40.35	46.55	82.09	78.90	80.39
RoBERTa-TopNeg‡	81.07	80.23	80.55	52.30	53.55	52.86	**60.55**	45.33	51.78	-	-	-
$\text{Conf-MPU}_{\text{LBiLSTM}}$	77.39	**82.84**	80.02	-	-	-	-	-	-	76.63	83.82	80.07
$\text{Conf-MPU}_{\text{BERT}}$	78.58	79.75	79.16	-	-	-	-	-	-	69.79	86.42	77.22
$\text{MProto}_{\text{BERT}}$	79.80	79.37	79.58	-	-	-	-	-	-	77.53	85.84	81.47
$\textbf{Ours}_{\text{BERT}}$	87.29	80.40	83.70	60.75	56.71	58.65	58.75	40.25	47.74	80.02	83.06	**81.50**
$\textbf{Ours}_{\text{RoBERTa}}$	**88.41**	82.77	**85.50**	**60.94**	**59.31**	**60.01**	58.22	**51.35**	**54.52**	-	-	-

Table 2: Comparisons with baselines on four datasets. † marks the results retrieved from (Zhou et al., 2022) and ‡ marks the results from (Xu et al., 2023). The best results are in **bold**.

Training Data	P	R	F1
$\mathcal{D}'$	88.41	82.77	85.50
$\mathcal{D}'_{\text{neg}}$	75.04	83.76	79.16
$\mathcal{D}'_{\text{pos}}$	93.03	64.94	76.49
$\mathcal{D}$	90.16	65.88	76.13

Table 3: Results of only cleaning either negative samples or positive samples. $\mathcal{D}'_{\text{pos}}$ and $\mathcal{D}'_{\text{neg}}$ denote denote only clean positive and negative, respectively. $\mathcal{D}'$ denotes data cleaned both and $\mathcal{D}$ is the original distant data.

6.1 Ablation Study

We conduct the ablation study on two aspects: ❶ only cleaning either negative samples or positive samples; ❷ varying the percentile in threshold samples for computing positive and negative thresholds. All results are reported from experiments on cleaned datasets with training dynamics from RoBERRTa-TopNeg for cleaning. Due to space limitations, we provide the analysis of CoNLL03 in this section. The ablation studies for the remaining datasets are included in Appendix F.

Cleaning positive/negative only. Table 3 shows the performance when only cleaning either negative samples or positive samples. We observe that if we single out the negative samples in cleaning, we improve the recall by a large margin, but with a notable loss in precision. Conversely, focusing on cleaning positive samples yields high precision but very low recall. This demonstrates the necessity of simultaneously removing both types of noisy annotation to create better distant labels.

Varying threshold sample percentile. The optimal threshold should eliminate as many false samples as possible while minimizing the removal of true samples. Figure 3 presents the performance and the number of discarded samples associated with various percentiles. The left plot illustrates the results when the percentile for computing the negative sample threshold is fixed at 90%, and the percentile for positive threshold samples is varied. The right plot shows the trends when the percentile of positive threshold samples is fixed at 100%, and the negative sample percentile is varied. For positive samples, both the precision and F1 scores show a consistent improvement. Regarding negative samples, though the recall continues to increase, the F1 score decreases when the percentile exceeds 90%. We also note that the number of discarded negative samples has a more pronounced increase as the percentile increases, which gives a significantly higher removal. The above empirical observation suggests that when the percentile exceeds 90% denoising becomes too aggressive in eliminating the false negatives which comes at the expense of discarding a large number of true negatives, leading to a decline in the overall performance. The true samples that exhibit training dynamics similar to false samples, are deemed *hard* samples (Talukdar et al., 2021).

6.2 Effective of Noise Cleaning

We conducted a further study to evaluate the noise cleaning effectiveness of our proposed DynClean. We use the original DS version and cleaned DS version of $\mathcal{D}'_4$ of CoNLL03 to train the RoBERTa and plot the class-wise performance curve on de-

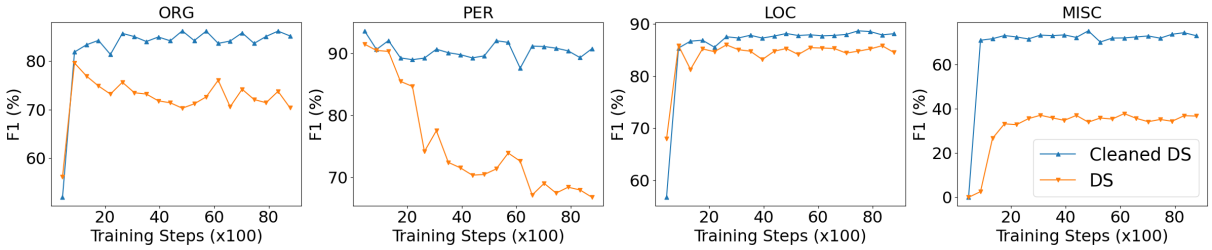

Figure 2: The performance curve for each class of CoNLL03 when training with the original DS and cleaned DS $\mathcal{D}'_4$ and testing on the dev set. "ORG", "PER", "LOC", and "MISC" represent the entity types of organization, person, location, and miscellaneous, respectively.

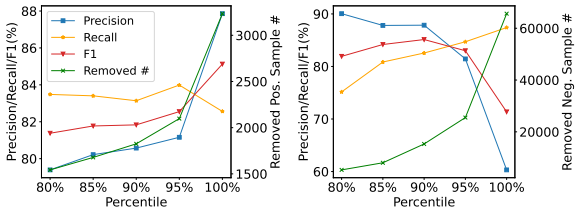

Figure 3: Ablation of varying percentile in threshold samples to compute the thresholds.

	CoNLL03	WikiGold	WNUT16	BC5CDR
AUM	84.72	59.88	54.45	81.42
Conf.	83.35	60.55	54.62	81.16

Table 4: Comparative F1-score results of AUM-based and Confidence-based training dynamics.

velopment set, as shown in Figure 2. We found that the model trained on our cleaned DS data consistently outperforms across classes than the model trained on original DS data, though we used less (but of higher-quality) training data. For the ORG and PER, we observed a gradual decrease in the model's performance when trained on the original DS dataset. This decline is attributed to the model overfitting on the extensive amount of noisy data. In contrast, training on the cleaned dataset allows the model to converge better. Notably, we find a significant improvement in the MISC. We provide in-depth for it in Appendix G.1. Specifically, since the MISC has the lowest proportion and the DS annotation results a large number of false negative annotation in MISC (i.e., missing MISC spans). Our cleaning method removed a substantial number of false negative spans of MISC, resulting in a significant performance improvement. We provide qualitative examples for our cleaning approach on CoNLL03 dataset in Appendix G.2. We provide additional noise cleaning analysis on the WikiGold dataset in Appendix G.3, as it represents a more complex DS-NER scenario.

6.3 Other Training Dynamics Metric

We evaluate another frequently used training dynamic metrics defined as mean (Confidence) and standard deviation (Variability) of the probabilities estimated by the model for a given label across epochs (Swayamdipta et al., 2020). We provide detailed definitions of both in Appendix H.1. The data are then categorized as **easy-to-learn**, **ambiguous**, and **hard-to-learn** according to the two metrics. It is hypothesized that the **hard-to-learn** region usually contains mislabeled samples (Swayamdipta et al., 2020). We apply the same threshold estimation strategy as proposed in Section 3.2 to estimate the Confidence thresholds for filtering out noisy samples. Specifically, the Eq. 9 in Algorithm 1 is replaced by Eq. 10. Table 4 presents the experimental results for the two metrics by using RoBERTa-TopNeg as the base model for both cleaning and evaluating. We observe that the two metrics exhibit very similar performance. This is because samples with low AUM values also have relatively low confidence scores, indicating they are **hard-to-learn** samples. Figure 9 in Appendix H.2 demonstrates a significant overlap between samples with low AUM values and those with low confidence values.

7 Conclusion

In this paper, we propose DynClean, a training dynamics-based cleaning approach for distantly supervised NER tasks. Unlike most existing methods that focus on learning from noisy labels, DynClean aims to improve the quality of data generated by distant supervision annotation. DynClean leverages the model behavior on each sample during the training to characterize samples, thereby locating both false positive and false negative annotations. Extensive experiment results show that models trained on our cleaned datasets achieve improvement ranges from 3.19% to 8.95% in F1-score and outperforms numerous advanced DS-NER works by significant margins up to 4.53% F1-score despite using fewer samples in training.

Limitations

Our method employs a span-based NER model, which has lower inference efficiency compared to token-based NER models. Although our proposed DynClean method achieves better performance than more sophisticated approaches, DynClean has high performance requirement on the model used for calculating accurate training dynamics, which increases the computational cost. Considering the effectiveness of performance improvement, we believe the additional computational costs are acceptable. Developing strategies to reduce the performance requirements for cleaning will increase the applicability of our method. Additionally, While our method successfully identifies numerous false samples, it may also inadvertently discard correctly labeled but hard samples. A future research direction involves refining our approach to better distinguish between false and hard samples. Furthermore, employing a noise-robust loss function on identified "noisy" samples may enhance model performance than simply removing them.

Ethical Statement

We comply with the ACL Code of Ethics.

References

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, Suntec, Singapore. Association for Computational Linguistics.

Shivani Choudhary, Niladri Chatterjee, and Subir Saha. 2023. IITD at SemEval-2023 task 2: A multi-stage information retrieval approach for fine-grained named entity recognition. In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 800–806, Toronto, Canada. Association for Computational Linguistics.

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

Jason Fries, Sen Wu, Alex Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *arXiv preprint arXiv:1704.06360*.

Fréderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle. 2015. Multimedia lab @ ACL WNUT NER shared task: Named entity recognition for Twitter microposts using distributed word representations. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 146–153, Beijing, China. Association for Computational Linguistics.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Zhanming Jie, Pengjun Xie, Wei Lu, Ruixue Ding, and Linlin Li. 2019. Better modeling of incomplete annotations for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 729–734, Minneapolis, Minnesota. Association for Computational Linguistics.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

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

Yangming Li, Lemao Liu, and Shuming Shi. 2022. Rethinking negative sampling for handling missing entity annotations. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7188–7197, Dublin, Ireland. Association for Computational Linguistics.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 1054–1064, New York, NY, USA. Association for Computing Machinery.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke

Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Zhiyuan Ma, Jintao Du, and Shuheng Zhou. 2023. Noise-robust training with dynamic loss and contrastive learning for distantly-supervised named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10119–10128, Toronto, Canada. Association for Computational Linguistics.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10367–10378, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Huitong Pan, Qi Zhang, Eduard Dragut, Cornelia Caragea, and Longin Jan Latecki. 2023. DMDD: A Large-Scale Dataset for Dataset Mentions Detection. *Transactions of the Association for Computational Linguistics*, 11:1132–1146.

Seo Yeon Park and Cornelia Caragea. 2022. A data cartography based MixUp for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4244–4250, Seattle, United States. Association for Computational Linguistics.

Minlong Peng, Xiaoyu Xing, Qi Zhang, Jinlan Fu, and Xuanjing Huang. 2019. Distantly supervised named entity recognition using positive-unlabeled learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2409–2419, Florence, Italy. Association for Computational Linguistics.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. 2020. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R Voss, and Jiawei Han. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 995–1004.

Aviad Sar-Shalom and Roy Schwartz. 2023. Curating datasets for better performance with example training dynamics. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10597–10608, Toronto, Canada. Association for Computational Linguistics.

Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning named entity tagger using domain-specific dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2054–2064, Brussels, Belgium. Association for Computational Linguistics.

Shuzheng Si, Zefan Cai, Shuang Zeng, Guoqiang Feng, Jiaxing Lin, and Baobao Chang. 2023. SANTA: Separate strategies for inaccurate and incomplete annotation noise in distantly-supervised named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3883–3896, Toronto, Canada. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Arka Talukdar, Monika Dagar, Prachi Gupta, and Varun Menon. 2021. Training dynamic based data filtering may not work for NLP datasets. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 296–302, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2022. Memorisation versus generalisation in pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7564–7578, Dublin, Ireland. Association for Computational Linguistics.

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

Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wiegers, and Zhiyong Lu. 2016. Assessing the state of the art in biomedical relation extraction: overview of the biocreative v chemical-disease relation (cdr) task. *Database*, 2016:baw032.

Shuhui Wu, Yongliang Shen, Zeqi Tan, Wenqi Ren, Jietian Guo, Shiliang Pu, and Weiming Lu. 2023. MProto: Multi-prototype network with denoised optimal transport for distantly supervised named entity

recognition. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2361–2374, Singapore. Association for Computational Linguistics.

Lu Xu, Lidong Bing, and Wei Lu. 2023. Sampling better negatives for distantly supervised named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4874–4882, Toronto, Canada. Association for Computational Linguistics.

Huaiyuan Ying, Shengxuan Luo, Tiantian Dang, and Sheng Yu. 2022. Label refinement via contrastive learning for distantly-supervised named entity recognition. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2656–2666, Seattle, United States. Association for Computational Linguistics.

Wenkai Zhang, Hongyu Lin, Xianpei Han, Le Sun, Huidan Liu, Zhicheng Wei, and Nicholas Yuan. 2021a. Denoising distantly supervised named entity recognition via a hypergeometric probabilistic model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14481–14488.

Xinghua Zhang, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Jiawei Sheng, Xue Mengge, and Hongbo Xu. 2021b. Improving distantly-supervised named entity recognition with self-collaborative denoising learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1518–1529, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xinghua Zhang, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Jiawei Sheng, Xue Mengge, and Hongbo Xu. 2021c. Improving distantly-supervised named entity recognition with self-collaborative denoising learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10746–10757, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kang Zhou, Yuepei Li, and Qi Li. 2022. Distantly supervised named entity recognition via confidence-based multi-class positive and unlabeled learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7198–7211, Dublin, Ireland. Association for Computational Linguistics.

A Datasets statistics

Table 5 presents the statistics of the used four datasets. "# Sent." denotes the number of sentences and "# Entity" indicates the number of entities (positive span samples) in the datasets. All the training sets are annotated by distant supervision, and the development and test sets are human annotated. The entity types number of CoNLL03, WikiGold, WNUT16, and BC5CDR are 4, 4, 10, and 2. CoNLL03 and WikiGold are the general domain NER. WNUT16 is open domain NER dataset and BC5CDR is biomedical domain NER dataset.

Datasets		CoNLL03	WikiGold	WNUT16	BC5CDR
Train	# Sent.	14,041	1,142	2,393	4,560
	# Entity	17,781	2,282	994	6,452
Dev	# Sent.	3,250	280	1,000	4,579
	# Entity	5,942	648	661	9,591
Test	# Sent.	3,453	274	3,849	4,797
	# Entity	5,648	607	3,473	9,809

Table 5: Statistics of four DS-NER datasets.

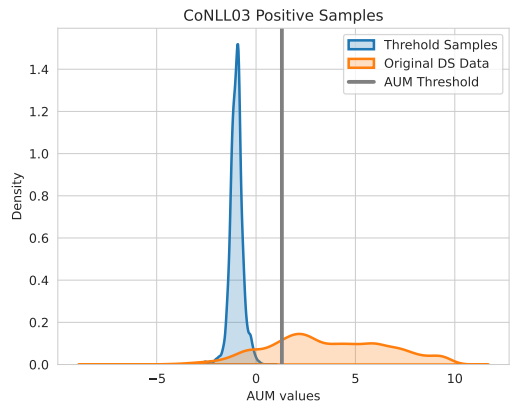

Figure 4: AUM distributions of positive samples and positive threshold samples.

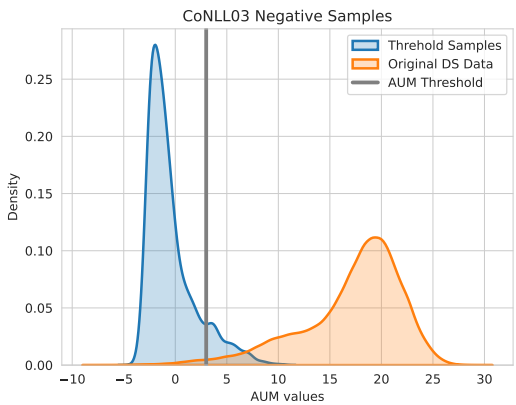

Figure 5: AUM distributions of negative samples and negative threshold samples.

Dataset	AUM k %ile in threshold samples		AUM %ile in DS data	
	Positive	Negative	Positive	Negative
CoNLL03	100%	90%	26.7%	1.5%
WikiGold	100%	90%	28.3%	0.5%
WNUT16	100%	90%	9.8%	0.1%
BC5CDR	90%	90%	1.3%	0.9%

Table 6: The percentile of the threshold values obtained through our proposed method in the threshold samples, and the corresponding percentile of the threshold values in the DS data.

B Effectiveness of threshold estimation strategy

Estimating the thresholds using threshold samples proves more consistent than direct tuning on the original DS data. Figure 4 and 5 illustrates the distribution of AUM values for both threshold samples and original DS samples in the CoNLL03 dataset. The gray lines in the figure represent the AUM threshold values corresponding to the k_{pos} and k_{neg} percentiles in threshold samples. Subsequently, samples with values lower than these threshold values (i.e., below the gray lines) in the original DS-NER dataset are eliminated during model training. In this case, the tuned percentiles are 100% for positive and 90% for negative in threshold samples. However, if tuning is performed directly on the original DS-NER dataset, the corresponding percentiles are 26.7% for positive and 1.5% for negative. Compared to those in threshold samples, which mimic the training dynamics of mislabeled samples, these percentiles are more challenging to obtain. Table 6 shows the corresponding percentiles of threshold values in threshold samples and original DS samples for all tested datasets. It is observed that the percentile selection in our method exhibits greater consistency. In contrast, the corresponding percentiles in the original DS-NER datasets are very dataset-dependent. This variation arises due to the differing noise distributions in DS-NER datasets, which complicates the direct tuning of percentile choices in the original datasets.

C Additional Experimental Settings

C.1 Additional Experimental Settings of DynClean

For a fair comparison, we use the same versions of the encoders in the above baselines. *bert-base-cased*[2] as BERT, *roberta-base*[3] as RoBERTa,

[2]https://huggingface.co/bert-base-cased
[3]https://huggingface.co/roberta-base

Please identify Person, Organization, Location and Miscellaneous Entity from the given text.

Text: All four teams are level with one point each from one game.
Entity:

Figure 6: The prompts and input formats for our experiments. The shown example is based on the entity types of CoNLL03 dataset.

and *biobert-base-cased-v1.1*[4] as BioBERT. For all tested span-based NER models, we use the same combnition of hyperparameters for all datasets: the learning rate is set as 1e-5; the training batch size is 16, and the maximum span length L is set as 8; 2 layers of feed-forward neural networks (FFNN) is employed as the classifier and the hidden size is set as 150, and the dropout rate is set as 0.2; the learnable width embedding size is 150. When using the TopNeg, we set the top-N_r as 5% as suggested by (Xu et al., 2023). For CoNLL03, BC5CDR, the training epoch E in 1 is 5. For WikiGold and WNUT16, the training epoch E is 10. Except for AutoNER, all other DS-NER baselines use the dev set for hyperparameter tuning. We follow this setting to tune the k_{neg} and k_{pos} for each dataset. Experiments were conducted using a single NVIDIA RTX 8000 GPU card with PyTorch 2.10.0, and the reported results represent the average of five runs, each with a different random seed. The average running time on the CoNLL03 dataset is 78 seconds/epoch.

C.2 Experimental Settings of LLMs

We implement the ChatGPT (*gpt-4-turbo*) and LLama-3.1-70B with zero-shot setting as in the distantly supervised NER task without any access to human annotated training label. The Llama-3.1-70B is ran on two A100 80G GPUs and uses the exactly same prompts of ChatGPT. For each dataset, we randomly sampling 200 samples from the text set each time. Our experiments were repeated five times and the results were averaged. We follow the prompt setting used in (Qin et al., 2023) as illustrated in Figure 6.

D Impact of TopNeg on Human Annotated Data

In Table 7, we show the results of training the span-based NER models with and without TopNeg on the Human-annotated (HA), Distantly Supervised (DS), and our best-cleaned CoNLL03 ($\mathcal{D}'_4$) datasets. We can observe that better performance is obtained when not using TopNeg on HA data. When training models on our denoised DS data, without TopNeg perform better as well. But TopNeg improves the models on the DS data. This indicates that our distant label cleaning method removes a large enough amount of false negatives.

[4]https://huggingface.co/dmis-lab/biobert-v1.1

Training Data	Model	P	R	F1
HA	BERT	89.71	90.32	90.01
	BERT-TopNeg	84.25	92.23	88.06
	RoBERTa	90.05	92.48	91.25
	RoBERTa-TopNeg	87.81	92.74	90.21
DS	BERT	89.75	63.08	72.09
	BERT-TopNeg	79.42	79.77	79.58
	RoBERTa	90.16	65.88	76.13
	RoBERTa-TopNeg	82.95	78.56	80.7
Ours	BERT	87.29	80.40	83.70
	BERT-TopNeg	85.71	81.16	83.37
	RoBERTa	87.64	82.67	85.08
	RoBERTa-TopNeg	86.34	83.17	84.72

Table 7: Comparisons on Human-annotated (HA), Distantly Supervised (DS), and our cleaned data of CoNLL03 for training.

	P	R	F1
Llama-3.1-70B	71.08	83.37	76.74
RoBERTa	86.84	76.04	81.07
Ours$_{\text{RoBERTa}}$	92.41	85.05	88.58

Table 8: Results of DynClean and RoBERTa trained on the distant labels from Llama-3.1-70B of CoNLL03 dataset.

E LLMs as Distant Annotator

We used the predictions of the open-source Llama-3.1-70B on CoNLL03 as distant labels. We follow the experimental settings in Section 5.2 to train RoBERTa and our DynClean using this data. The experimental results are shown in Table 8. We found that, because Llama-3.1-70B outperformed KB-Matching, our model achieved further improvements. This demonstrates the feasibility of combining our method with LLM predictions as distant labels to enhance the DS-NER performance.

F Ablation Study on Other Datasets

Cleaning positive/negative only. Table 9 shows the performance when only cleaning either negative samples or positive samples for each remaining dataset. The results demonstrate the necessity of simultaneously removing both types of noisy annotation to create better distant labels for each dataset.

Training Data	WikiGold			WNUT16			BC5CDR		
	P	R	F1	P	R	F1	P	R	F1
$\mathcal{D}'$	60.94	59.31	60.09	58.22	51.35	54.52	80.02	83.06	81.50
$\mathcal{D}'_{\text{neg}}$	52.78	60.96	56.58	56.24	48.43	52.04	76.45	84.29	80.18
$\mathcal{D}'_{\text{pos}}$	59.16	55.35	57.19	62.54	45.24	52.49	90.76	66.92	77.04
$\mathcal{D}$	60.63	44.65	51.42	62.72	43.45	51.34	89.44	67.57	76.98

Table 9: Results of only cleaning either negative samples or positive samples. $\mathcal{D}'$ is the cleaned dataset with both and $\mathcal{D}$ is the original distant data. $\mathcal{D}'_{\text{pos}}$ and $\mathcal{D}'_{\text{neg}}$ denote cleaned datasets only consider positive samples and negative samples, respectively.

Varying threshold sample percentile. Figure 7 presents the performance and the number of discarded samples associated with various percentiles on the remaining datasets. The results shows the k_{pos} and k_{neg} are relative consistent across different datasets.

G Additional Analysis

G.1 Effectiveness of Cleaning

To further understand the effectiveness of our cleaning method, we using the human annotated CoNLL03 data as ground truth to annotated labels in DS and Cleaned DS $\mathcal{D}'_4$, respectively. We then compute the number true positive labels and false labels (both false negative and false positive) in in DS and Cleaned DS. We provide the results in Table 10. In the Table 10, the "# Pos." denotes the correct positive labels for each class. "# False Neg. + # False Pos." are two types of noisy samples, where the "# False Pos." refers to incorrect labels for each positive class (e.g., a PER span is labeled as LOC span or non-Entity span), "# False Neg." refers to an entity span missed by distant supervision (e.g., a PER entity is not in the knowledge base, but exists in the text). Considering both is because the DS-NER performance is impacted by two kinds of noise, i.e., false negatives and false positives.

We can find that our methods signficantly reduce the total number of false annotations in original DS data. This is also why our method can improve the performance. Particularly, we notice that there is a large number of missing annotations of MISC class, i.e., 2541 false negatives, in the distant supervision step. Thus, the small number of positives and the large number of false negatives cause the model to overfit when trained on the original DS data. After cleaning, we significantly decrease the number of false annotations, resulting the performance increases from 29.96 to 73.20 F1 score as shown in Figure 2.

		ORG	MISC	PER	LOC
DS	# Pos.	4128	786	7535	5327
	# True Pos.	3754	786	4782	5326
	# False Pos. + False Neg.	2668	2541	4565	1500
Cleaned DS	# Pos.	3387	709	3933	5070
	# True Pos.	3241	709	3446	5069
	# False Pos. + False Neg.	293	481	552	388

Table 10: The number of positive labels, the number of true positives, and the corresponding number of false positives and false negatives for each entity type. The results are obtained by comparing the human annotated labels with DS and Cleaned DS $\mathcal{D}'_4$, respectively.

G.2 Qualitative Examples

To give an intuition of the benefits of our proposed false annotation cleaning approach for DS-NER, we present qualitative examples in Figure 11. It has two examples. One observes that compared to the HA (Human-Annotated) instance, the DS (Distant Supervision) instance omits the entity spans "PSA", "Orioles", and "Andy Etchebarren" (false negatives), incorrectly labels "Union" and partially labels "Andy" as entity spans (false positive). Our method successfully identifies both false negative and false positive annotations in the DS data. Avoiding such false samples for training can alleviate the models' overfitting noise. This also indicates that our method can assist the distant annotation process and improve the data quality.

G.3 Effectiveness of Noise Cleaning on WikiGold

We conduct similar noise cleaning analysis on WikiGold, as it has lower KB-Matching performance. We using the original DS and cleaned DS $\mathcal{D}'_4$ of WikiGold to train the RoBERTa and plot the class-wise performance curve on dev set, as shown in Figure 8. We find that the model trained on our cleaned data consistently outperforms across classes than the model trained on original DS data, though we used less (but of higher-quality) training data.

Similar to Appendix G.1, we compute the number of true positive labels and false labels (both false

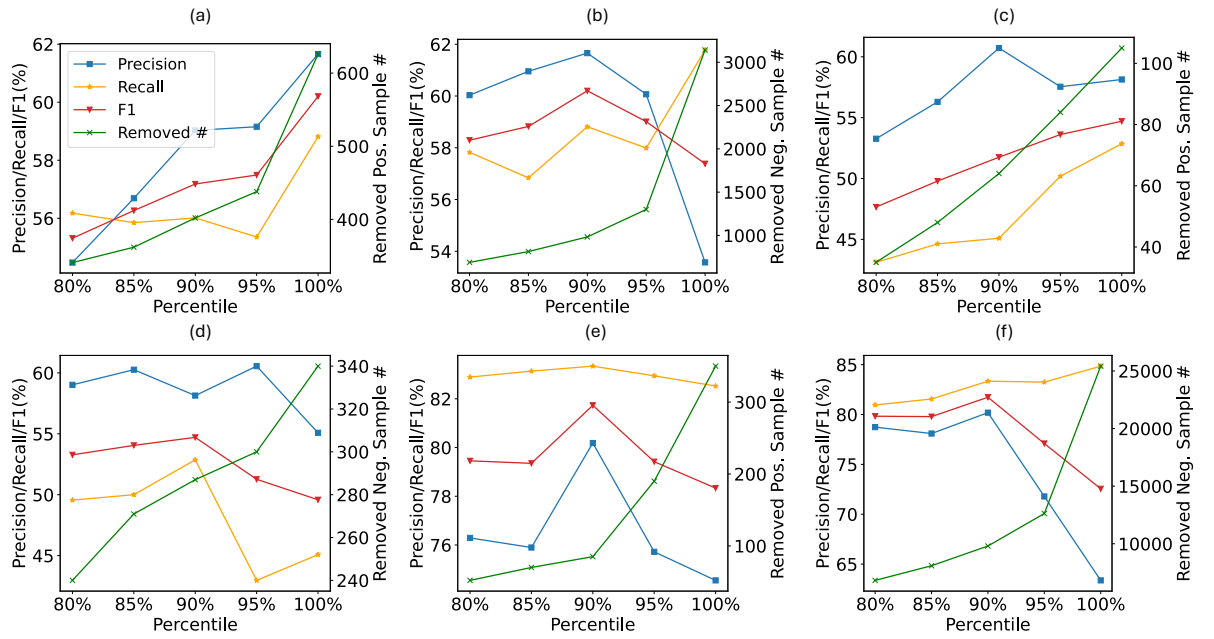

Figure 7: Ablation of varying percentile in threshold samples to compute the thresholds on different datasets. Subfigure (a) and (b) represent the WikiGold dataset, subfigure (c) and (d) represent the WNUT16 dataset, and subfigure (e) and (f) represent the BC5CDR dataset.

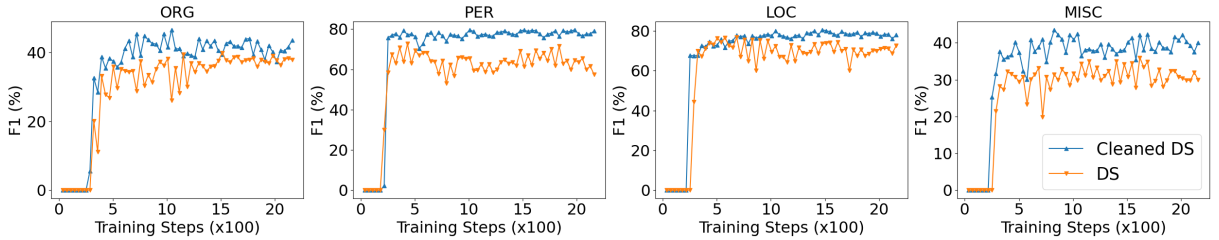

Figure 8: The performance curve for each class of WikiGold when training with the original DS and cleaned DS $\mathcal{D}'_4$ and testing on the dev set. "ORG", "PER", "LOC", and "MISC" represent the entity types of organization, person, location, and miscellaneous, respectively.

negative and false positive) in in DS and Cleaned DS. Table 12 shows the detailed cleaning effectiveness. We can find that our method also signficantly reduces the total number of false annotations in original DS data. Thus, the models trained our cleaned dataset achieve better performances.

		ORG	MISC	PER	LOC
DS	# Pos.	715	440	704	421
	# True Pos.	240	159	358	335
	# False Pos. + False Neg.	745	529	567	325
Cleaned DS	# Pos.	511	382	399	365
	# True Pos.	195	143	324	322
	# False Pos. + False Neg.	507	418	176	138

Table 12: The number of positive labels, the number of true positives, and the corresponding number of false positives and false negatives for each entity type. The results are obtained by comparing the human annotated labels with DS and Cleaned DS $\mathcal{D}'_4$, respectively.

H Other Training Dynamic Metric

H.1 Definition of Confidence and Variability

The "Confidence" score is defined as the mean model probability of the assigned label y^* (potential error) across epochs:

$$\hat{\mu}(\mathbf{x}, y^*) = \frac{1}{E} \sum_{e=1}^{E} P(\mathbf{x}, y^*) \tag{10}$$

Where P denotes the model's probability at the end of eth epoch during training. The "Variability" is defined as the standard deviation of P across epochs E:

$$\hat{\sigma} = \sqrt{\frac{\sum_{e=1}^{E} \left(P(y^*|\mathbf{x}) - \hat{\mu}(\mathbf{x}, y^*)\right)^2}{E}} \tag{11}$$

These two metrics are used to evaluate the characteristics of each individual sample. When using

DS instance: [Union]$_{ORG}$ officials from the [Public Service Association]$_{ORG}$ (PSA) were unavailable for comment.	
HA instance: Union officials from the [Public Service Association]$_{ORG}$ ([PSA]$_{ORG}$) were unavailable for comment.	
Ours: [~~Union~~]$_{FP}$ officials from the [Public Service Association]$_{ORG}$ ([~~PSA~~]$_{FN}$) were unavailable for comment.	
DS instance: Orioles's bench coach [Andy]$_{PER}$ Etchebarren will manage the club in [Johnson]$_{PER}$'s absence.	
HA instance: [Orioles]$_{ORG}$'s bench coach [Andy Etchebarren]$_{PER}$ will manage the club in [Johnson]$_{PER}$'s absence.	
Ours: [~~Orioles~~]$_{FN}$'s bench coach [[~~Andy~~]$_{FP}$ ~~Etchebarren~~]$_{FN}$ will manage the club in [Johnson]$_{PER}$'s absence.	

Table 11: Case study for the CoNLL03 dataset. "DS" denotes the distantly annotated instances and "HA" denotes the Human-Annotated instances. The FN and FP are the identified false negative and false positive annotations in DS data by our method, which are removed during training.

in our DynClean, we directly replace the 9 with 10 in Algorithm 1.

H.2 Data Map Visulization of AUM and Confidence

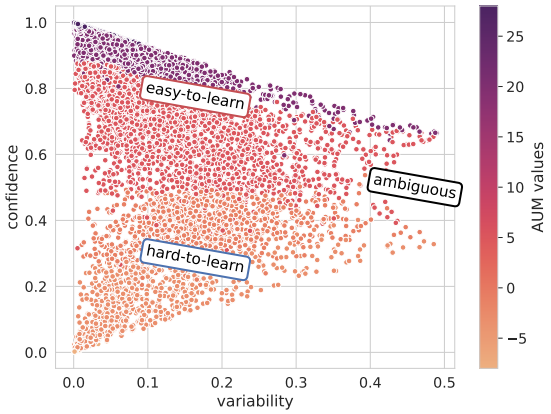

Figure 9: Data Map of the positive samples in CoNLL03. We can find that the samples with lower AUM values are mainly in the **hard-to-learn** region.

We visualize the AUM values with their data map as defined by (Swayamdipta et al., 2020) in Figure 9 for positive samples of CoNLL03. We observe that existing a significant overlap between samples with low AUM values and those with low Confidence values. This futher shows that the two metrics have very similar effects on identify false samples.