Named Entity Recognition through Deep Representation Learning and Weak Supervision

Jerrod Parker
The Vanguard Group
University of Toronto
jerrod_parker@vanguard.com

Shi Yu
The Vanguard Group
shi_yu@vanguard.com

Abstract

Weakly supervised methods estimate the labels for a dataset using the predictions of several noisy supervision sources. Many machine learning practitioners have begun using weak supervision to more quickly and cheaply annotate data compared to traditional manual labeling. In this paper, we focus on the specific problem of weakly supervised named entity recognition (NER) and propose an end-to-end model to learn optimal assignments of latent NER tags using observed tokens and weak labels provided by labeling functions. To capture the sequential dependencies between the latent and observed variables, we propose a sequential graphical model where the components are approximated using neural networks. State-of-the-art contextual embeddings are used to further discriminate the quality of noisy weak labels in various contexts. Results of experiments on four public weakly supervised named entity recognition datasets show a significant improvement in F1 score over recent approaches.

1 Introduction

Many industries and organizations have collected large amounts of unlabeled text data that they want to make use of for various Natural Language Processing (NLP) applications. However, in many of these applications (named-entity recognition, question answering, text summarization, relation extraction), obtaining a large number of labels can be prohibitively expensive, error-prone, or otherwise infeasible. Furthermore, domain adaptation (Han and Eisenstein, 2019), which is commonly used in scarce label settings, can often struggle in new emerging / specific domains that don't have any closely related labeled datasets.

Under the absence of a closely related labeled dataset, *weakly supervised learning* is often used as a cheaper, less time-consuming alternative to obtaining gold standard labels. The main idea of weak supervision is to approximate the true labels by integrating multiple sets of noisy training labels. Each set of noisy labels (commonly referred to as "weak labels") is provided by a weak labeling function which often comes in the form of a knowledge base, heuristic, or pre-trained model. Weak supervision has obtained a lot of success in several NLP tasks containing approximately *i.i.d.* data such as topic classification (Bach et al., 2019), sentiment analysis, and social media content tagging (Fu et al., 2020).

Weak supervision on sequential data labeling problems such as NER is an emerging topic. Most current methods require either the time-consuming creation of additional heuristics such as 'linking rules'(Safranchik et al., 2020) or 'entity boundary detectors'(Fries et al., 2017), or assume that the accuracy of a weak labeler only depends on the true latent class (Safranchik et al., 2020; Lison et al., 2020; Fries et al., 2017). The latter is likely suboptimal since we would expect the accuracy to vary even within instances of the same class depending on the context given by surrounding tokens. One exception to this is the Fuzzy-LSTM-CRF(Shang et al., 2018). However, it ignores which weak labeler each prediction came from as well as the number of predictions for each class. This could be an issue when the weak labelers have differing accuracies, since the model may learn from a majority of wrong labels.

To address these foregoing issues, one of our main contributions is the proposal of an end-to-end method called **D**eep **W**eak Supervision on **S**equential Data (DWS), that learns context-dependent proficiency representations for weak labelers, enhanced further through contextual embeddings from pre-trained language models. In addition, instead of following the traditional approach that treats named entity (NE) tags as multi-

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3828–3839

nomial samples, we directly model the conditional dependency of tags on tokens as interactions of two representations: tag representation and contextual token representation.

Advantages of adopting representation learning are two-fold: First, learning context-aware proficiency of labeling functions enables the weak supervision procedure to denoise unreliable labels at tokens where labeling functions have many disagreements. Second, because embedding methods have demonstrated great potential in capturing semantic meanings, the proposed model allows flexible transfer of existing NER pipelines to new domains through leveraging pre-trained domain-specific embeddings.

DWS relies on a graphical model to capture statistical dependencies among tokens, weak labels and true latent NE tags. However, latent variable estimation is challenging and the techniques are often both sample and computationally complex. For example, Ratner et al. (2017) required a Gibbs-based algorithm, and Ratner et al. (2019) required estimating the full inverse covariance matrix among the labelers. In video analysis, Varma et al. (2019b) required the use of multiple iterations of stochastic gradient descent (SGD) to learn accuracy parameters, but the dependencies are limited to weak labels and true labels. When context of the sequential inputs are directly involved in the model, optimizing or even formulating the analytical solution becomes much more difficult.

Our solution is motivated by the advantages of using neural networks to model the transition and output distributions (Bengio and Frasconi, 1996; Li and Shum, 2006). Instead of deriving analytical formulations to learn the parameters for the given structure, we use deep neural networks to approximate conditional dependencies and sequential transitions. Furthermore, the marginal likelihood of the proposed model is optimized via hard EM (Min et al., 2019) to find the most probable sequence of latent tags. Such a hybrid process allows us to easily integrate distributed representations in our model and also enables us to explore complex model structures.

We benchmarked the proposed DWS model on several NER datasets and compared it with some recent weak supervision approaches. Experimental results show DWS's advantage on tagging tasks where there are a lot of conflicting weak labels. Furthermore, we conduct detailed analysis to evaluate the complexity of these NER datasets in terms of the number of NE tags and labeling functions and also the amount of inconsistency among the weak labels.

We conclude that some datasets used in past works are useful to evaluate the robustness of weak supervision models, whereas others contain weak labels that are trivial to denoise and result in all algorithms looking equivalent. Therefore, directly comparing results on weakly labeled datasets without accounting for the difficulty of denoising their labels can drive misleading conclusions. To more rigorously test how the performance of the algorithms scale with the difficulty of the weak supervision problem, we introduce a new method which stratifies a dataset into tokens containing varying levels of weak labeler disagreement, quantified by the entropy of the weak label predictions, and compares how the performance of each algorithm scales with the difficulty of the denoising task. Results show that the performance advantage of our algorithm over the current methods grows quickly with respect to the disagreement among the weak labelers.

2 Related Work

Snorkel (Ratner et al., 2017) is a well-known tool that learns a generative model to estimate the accuracies and correlations between weak labelers on *i.i.d.* data. SwellShark (Fries et al., 2017) treats weakly supervised NER as an *i.i.d.* task but requires creating additional heuristics (sometimes called entity span generators) to find the entity boundaries. In recent years, various new algorithms have been introduced to extend the weak supervision idea to tasks involving sequential data such as NER. BOND (Liang et al., 2020) and AutoNER (Shang et al., 2018) respectively denoise predictions of a single weak labeler and a set of dictionaries. Several generative approaches such as Hidden Markov Model (Lison et al., 2020; Safranchik et al., 2020) and a discriminative method Fuzzy-LSTM-CRF (Shang et al., 2018) have been proposed to model the dependencies between NE tags; thus, entity span generators are no longer needed. There have also been several relevant approaches on other weak supervision tasks. ReHession (Liu et al., 2017) was developed for weakly supervised relation extraction and tries to learn the contexts where each weak labeler is proficient, then uses that knowledge to infer the true labels. Varma et al.

(2019a) proposes a robust PCA-based algorithm to learn dependency structures for image classification. In the application of video analysis, Fu et al. (2020) applies a general binary Ising model to factorize likelihood expectations over cliques so an analytical solution is found to speed up the model parameter learning.

In this paper, we systematically compare most of the recent approaches proposed for NER except for BOND, AutoNER, and Swellshark which are not directly applicable to our experimental settings.

3 Problem Definition

We assume a sequential labeling problem formulation where we are given a sequence of tokens $\mathcal{X} = \{x_1, ..., x_N\}$ which map to a sequence of latent class variables $\mathcal{Y} = \{y_1, .., y_N\}$. In a fully supervised scenario, $\mathcal{Y}$ is usually partially annotated thus $\mathcal{Y} = \mathcal{Y}_{\text{train}} \cup \mathcal{Y}_{\text{test}}$, so model parameters are estimated from ($\mathcal{X}_{\text{train}}$, $\mathcal{Y}_{\text{train}}$). Because $\mathcal{Y}_{\text{train}}$ is usually expensive to obtain, our primary goal is to estimate $\mathcal{Y}_{\text{train}}$ with the help of multiple, potentially noisy labeling sources. Suppose we have a set of weak labels $\mathcal{L}_{\text{train}}$ available for $\mathcal{X}_{\text{train}}$, where each token is assigned a set of weak labels provided by m different labeling sources $\lambda_1, ..\lambda_m$ (labelers can choose to make no prediction by voting 'Abstain'). For simplicity, we drop the notion of training and test data, and simply use $\mathcal{X}, \mathcal{Y}, \mathcal{L}$ to represent all tokens, latent tags, and weak labels in our problem. Additionally, we define the vote of weak labeler λ_j on token x_i as $l_{i,j}$.

In a supervised setting where $\mathcal{Y}$ are given, we learn model parameters Θ by maximizing the log-likelihood of $\mathcal{Y}$ given the input $\mathcal{X}$ with respect to Θ:

$$J_{sup}(\Theta|\mathcal{X}, \mathcal{Y}) = \max \log P(\mathcal{Y}|\mathcal{X}; \Theta) \quad (1)$$

whereas in our weak supervision scenario, $\mathcal{Y}$ are hidden and $\mathcal{L}$ are fully observable, so the learning objective can instead be to maximize the marginal likelihood of the weak labels. Letting $\mathcal{E}$ be the set of K entity classes which includes 'No Entity', we have:

$$P(\mathcal{L}|\mathcal{X}; \Theta) = \sum_{Y_c \in \mathcal{E}^N} P(\mathcal{L}, Y_c|\mathcal{X}; \Theta) \quad (2)$$

which can be used to compute the objective as follows:

$$J_{weak}(\Theta|\mathcal{X}, \mathcal{L}) = \max \log P(\mathcal{L}|\mathcal{X}; \Theta) \quad (3)$$

4 Model Overview

4.1 Model Structure and Likelihood

To incorporate the dependencies among tokens, weak labels, and true labels, we define a graphical structure illustrated as Figure 1. The proposed graphical model is partially directed and its analytical solutions and exact inference are available through segmentations of the input sequence but are usually complex to derive. In our approach, the likelihood $P(\mathcal{L}, \mathcal{Y}|\mathcal{X})$ defined in (2) is factorized as $P(\mathcal{L}|\mathcal{X}, \mathcal{Y})$ and $P(\mathcal{Y}|\mathcal{X})$, where each is approximated via a neural network. These networks are described in Sections 4.2 and 4.3, respectively. The parameters are learned through maximizing an approximation to the marginal likelihood $P(\mathcal{L}|\mathcal{X})$ using a hard EM algorithm. Following is an in depth description of the model formulation.

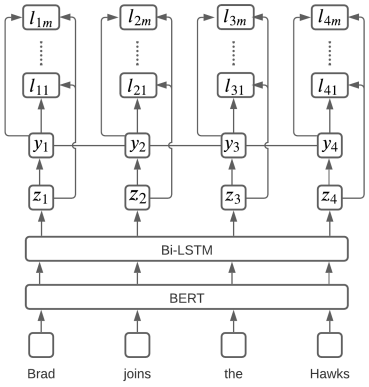

Figure 1: The DWS model where z_i denotes the contextual embedding for the token at index i.

According to (3) and (2) we have:

$$\begin{aligned} P(\mathcal{L}|\mathcal{X}) &= \sum_{Y_c \in \mathcal{E}^N} P(\mathcal{L}, Y_c|\mathcal{X}) \\ &= \sum_{Y_c \in \mathcal{E}^N} P(\mathcal{L}|\mathcal{X}, Y_c) P(Y_c|\mathcal{X}) \end{aligned}$$

We now assume different labeling functions are independent of each other given the input text and true labels. Additionally, we assume the weak labels at token x_i are conditionally independent of the true labels at any index $k \neq i$ when given the input text and true label at index i (d-separation details are in Appendix A.2). Therefore, we have:

$$P(\mathcal{L}|\mathcal{X}, Y_c) = \prod_i \prod_j P(l_{i,j}|\mathcal{X}, Y_c) \quad (4)$$

$$= \prod_i \prod_j P(l_{i,j}|\mathcal{X}, y_{c,i}) \quad (5)$$

where $y_{c,i}$ denotes the label at index i of Y_c. This

means:

$$\log P(\mathcal{L}|\mathcal{X}) = \log \sum_{Y_c \in \mathcal{E}^N} P(Y_c|\mathcal{X}) \prod_i \prod_j P(l_{i,j}|\mathcal{X}, y_{c,i}) \tag{6}$$

In it's current form, the conditional log likelihood is difficult to optimize because of the sum-product inside the logarithm. As an alternative, we maximize J_{HardEM} which is an approximation to the conditional log-likelihood where the summation over entity labels is replaced with a maximum.

$$\begin{aligned} J_{HardEM} &= \log \Big[\max_{Y_c \in \mathcal{E}^N} P(Y_c|\mathcal{X}) \prod_i \prod_j P(l_{i,j}|\mathcal{X}, y_{c,i}) \Big] \\ &= \max_{Y_c \in \mathcal{E}^N} \Big[\log P(Y_c|\mathcal{X}) + \sum_i \sum_j \log P(l_{i,j}|\mathcal{X}, y_{c,i}) \Big] \\ &= \max_{Y_c \in \mathcal{E}^N} \log P(\mathcal{L}, Y_c|\mathcal{X}) \end{aligned} \tag{7}$$

This new optimization problem attempts to find accurate modes of $P(\mathcal{L}, \mathcal{Y}|\mathcal{X})$. All that remains is to formulate $P(Y_c|\mathcal{X})$ and $P(l_{i,j}|\mathcal{X}, y_{c,i})$ for some pair (i, j).

4.2 Modeling Weak Labeler Representations

In practice, a weak labeler is usually derived from a specific rule or a controlled vocabulary. Thus, it is reasonable to assume its accuracy depends on the context of the token it is making a prediction on. Inspired by Liu et al. (2017), we model labeling function λ_j providing the correct label to x_i as a discrete event following a *Bernoulli* distribution, given by:

$$P(l_{i,j}|\mathcal{X}, y_{c,i}) = \begin{cases} \alpha_{ij} & \text{if } l_{i,j} = y_{c,i} \\ 1 - \alpha_{ij} & \text{if } l_{i,j} \neq y_{c,i} \end{cases} \tag{8}$$

Here, $\alpha_{ij} = \sigma(\frac{z_i^T \theta_j}{\sqrt{d}})$, σ is the sigmoid function, θ_j is a learnable embedding specific to λ_j, and $z_i (z_i \in \mathbb{R}^d)$ is the contextual embedding of the token x_i. This formulation improves the modeling capacity over many past methods such as Snorkel (Ratner et al., 2017) or the HMM (Safranchik et al., 2020; Lison et al., 2020) which have purely class-conditioned accuracies. In addition, it allows the utilization of external knowledge from large pre-trained language models by making z_i a function of their contextual embeddings. When a weak labeler abstains, there is no concept of accuracy. We chose to set the probability of the 'Abstain' votes to 1, but in the future could instead model the probability of abstaining.

4.3 Modeling Class Representations and Transition Scores

We model $P(Y_c|\mathcal{X})$ as a function of the contextual embeddings z. The model uses a linear-chain conditional random field (CRF) output layer which is often utilized to model dependencies between labels (Huang et al., 2015; Lample et al., 2016; Akbik et al., 2018). The distribution $P(Y_c|\mathcal{X})$ is given by

$$P(Y_c|\mathcal{X})) = \frac{e^{C(\mathcal{X}, Y_c)}}{\sum_{Y' \in \mathcal{E}^N} e^{C(\mathcal{X}, Y')}} \tag{9}$$

$$\text{where} \quad C(\mathcal{X}, Y) = \sum_{j=1}^{N} s_{j,y_j} + \sum_{j=1}^{N-1} T[y_j, y_{j+1}] \tag{10}$$

$$\text{and} \quad s_{j,y_j} = \frac{z_j^T t_{y_j}}{\sqrt{d}} \tag{11}$$

where T is a learnable matrix defining the transition scores between any two entity classes and t_{y_i} is the learnable embedding for class y_i. We have scaled the dot products in both the formula for α_{ij} and (11), by the square root of the embedding size d to significantly increase the stability of training as was shown to be useful in Vaswani et al. (2017).

4.4 Algorithm for Optimization

To maximize the objective J_{HardEM}, we repeat the following two steps:

1. Calculate $Y' = \arg\max_{Y_c \in \mathcal{E}^N} \Big(\log P(\mathcal{L}, Y_c|\mathcal{X}) \Big)$
2. Maximize $\log P(\mathcal{L}, Y'|\mathcal{X})$ for one gradient ascent step.

This approach is often termed *hard EM* and has been successful in other areas such as weakly supervised relation extraction (Liu et al., 2017) and question answering (Min et al., 2019). Step 1 can be computed as follows:

$$Y' = \arg\max_{Y_c \in \mathcal{E}^N} \Big(\log P(\mathcal{L}, Y_c|\mathcal{X}) \Big) \tag{12}$$

$$= \arg\max_{Y_c \in \mathcal{E}^N} \Big(\sum_{i=1}^{N} s'_{i,y_{c,i}} + \sum_{k=1}^{N-1} T[y_{c,k}, y_{c,k+1}] \Big) \tag{13}$$

$$\text{where} \quad s'_{i,y_{c,i}} = s_{i,y_{c,i}} + \sum_{j=1}^{m} \log P(l_{i,j}|\mathcal{X}, y_{c,i}) \tag{14}$$

The details of the derivation are in Appendix A.1. Equation (13) can be solved efficiently with the same Viterbi decoding algorithm used in Huang et al. (2015). In practice, we constrain each y'_i to be in the set of classes voted on among weak labelers on the token x_i by setting $s'_{i,y_{c,i}} = -\infty$ for

any $y_{c,i}$ that does not satisfy the constraint. Additionally, we found it useful to penalize choosing sequences Y' containing illegal class transitions. This was done by adding a hyperparameter τ to the transition matrix T from (13) in all locations that correspond to illegal transitions. For example, transitioning from the middle to the beginning of a 'Person' entity (i.e, I-Per $\rightarrow$ B-Per) would be penalized as it does not make sense.

4.5 Training Details

The contextual token embeddings z used to calculate (8) and (11) are obtained from a trainable single layer bidirectional LSTM that uses contextual word embeddings from pre-trained BERT models as described in Section 5.2. We choose to freeze the parameters of the pre-trained models to allow fast and inexpensive training.

To obtain a better starting point for the EM algorithm, the models are given a warm start by training for one epoch where the token labels Y' in step 2 of the optimization algorithm in Section 4.4 are set to be the majority vote labels.

Lastly, we choose the model parameters to be those that give the best entity F1 score on a held out validation set over 10 random restarts of 5 epochs each. Further details of the training procedure and architectures used are described in Appendix A.3 and A.4.

5 Experiments and Results

Experiments are conducted on four public weakly supervised NER datasets (Lison et al., 2020; Safranchik et al., 2020) as summarized in Table 1. We use the same weak labeling functions as reported in these approaches. Each dataset is split into train, validation, and test set using the same splits as Safranchik et al. (2020) suggests for NCBI-Disease, BC5CDR, and LaptopReview, and Liang et al. (2020) for CoNLL2003[1].

5.1 Weakly Labeled Dataset Difficulty

We first conduct a study to establish the difficulty of denoising the weak labels in each dataset. As we will demonstrate, this allows a detailed analysis of the strengths and weaknesses of each model. Both the amount and types of disagreement among the weak labelers differ drastically between weakly supervised NER datasets. Knowledge about the types of difficulties in each dataset is necessary to understand the contexts where each model is most useful as well as how to gauge which results are the most important. We categorize disagreements into three types: **Token Position** (i.e., $\{B, I, L, U, O\}$), **Positioned Class** (i.e., $\{B\text{-}Per, I\text{-}Per, L\text{-}Per\}$), and **Unpositioned Class** (i.e., $\{Per, Loc, Org, Misc\}$).

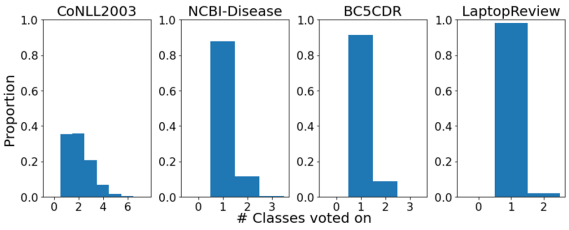

Figure 2: The tokens in each dataset containing weak labels were bucketed based on the number of classes voted on among weak labelers. This figure shows the proportion of tokens in each bucket.

To understand the prominent types of disagreement in each weakly labeled dataset, we calculated the number of tokens which contain each type of disagreement. The results are displayed in the last three columns of Table 1, and show that weak labelers on NCBI-Disease, BC5CDR, and LaptopReview disagree on a token's positioning within an entity much more often than on the unpositioned entity class. This indicates that the main difficulty on those weakly labeled datasets is resolving disagreements on where each entity begins and ends. To the contrary, CoNLL2003 contains many disagreements on both the token position and unpositioned class predictions.

We also include Figure 2 to emphasize the difficulty of the weakly labeled CoNLL2003 dataset compared to the others. On average there are a much higher number of classes being disagreed upon per token, a heuristic for the difficulty of the weakly labeled dataset.

5.2 Results

Our experiments focus on the following topics: (1) Comparison of the proposed method's performance with existing methods on benchmark datasets. (2) Stratified analysis and benchmarking of various models w.r.t. the difficulty of the NER task. (3) Systematic study to establish the importance of various design decisions of our proposed method.

The proposed DWS model is compared to sev-

[1] Lison et al. (2020) did not define a validation set or test set split. Instead, their results were on the full dataset.

Dataset	Document Type	#WL's	#Tokens	#Entities	Entity Types	#Unpositioned	#Positioned	#Position
CoNLL2003	News	46	301,649	35,089	Per,Loc,Org,Misc	7439	9236	6203
LaptopReview	Online reviews	12	63,379	3,005	Laptop Aspect	23	43	40
NCBI-Disease	PubMed Articles	12	183,330	6,873	Disease	310	923	904
BC5CDR	PubMed Articles	26	351,508	28,687	Chemical,Disease	216	1058	1001

Table 1: Summary statistics of the four datasets where WL refers to 'Weak Labelers'. The weak labels for CoNLL2003 are defined by Lison et al. (2020) and the weak labels for NCBI-Disease, LaptopReview, and BC5CDR are defined by Safranchik et al. (2020). The last three columns are defined as: **'#Unpositioned'**: is the number of tokens with more than one unpositioned class voted on. **'#Positioned'**: is the number of tokens with more than one positioned class voted on. **'#Position'**: is the number of tokens with more than one position (i.e, B,I,L,U) voted on. On CoNLL2003, the weak labelers were created such that their votes are in the set of the 19 OntoNotes5.0 classes. The weak supervision model is trained using this full set of classes and then at inference time, it's predictions are mapped to the set of CoNLL2003 classes or 'No Entity' for types such as Date or Ordinal.

eral recently introduced models which are trained on the benchmark datasets. Those include **Snorkel** (Ratner et al., 2017) where each token is considered as an independent example, **Fuzzy-LSTM-CRF** (Shang et al., 2018), **HMM** (Lison et al., 2020), the HMM formulation from Safranchik et al. (2020) which does not use 'linking rules', **Majority Vote**, and **Unweighted Vote** (Safranchik et al., 2020) which creates probabilistic training labels using the empirical distribution of the weak labels on each token. To make a fair comparison with DWS, the Fuzzy-LSTM-CRF uses the same model architecture as its discriminative component $P(\mathcal{Y}|\mathcal{X})$. To enhance performance, we use contextual token embeddings from RoBERTa (Liu et al., 2019) on CoNLL2003, uncased BERT (Devlin et al., 2019) on Laptop Review, and BioBERT (Lee et al., 2019) on NCBI-Disease and BC5CDR as inputs to our model and the Fuzzy-LSTM-CRF.

Our weak supervision pipeline has two steps: **Step 1)** First it learns the latent labels of the training data using weak labels and tokens, so the learned data can be used as annotated training data to train NER classifiers. We measure model performance in this step through evaluating the micro-entity precision, recall, and F1 scores of the learned latent labels against the ground truth labels. Table 2 (upper) displays these results and shows that the proposed DWS method creates training labels with an F1 score of 1.65% higher than the nearest comparing model on CoNLL2003 and F1 scores within 0.1% and 0.03% of the best comparative approach on BC5CDR and NCBI-Disease, respectively. Additionally, DWS is outperformed by an F1 score of 0.58% on LaptopReview but this is expected since the dataset only contains 43 tokens where the weak labelers disagree. This is likely far too little for DWS to learn robust deep representations for the accuracies of each weak labeler.

Step 2) Using the learned labels as training labels, we train classifiers and apply the trained classifiers on test data. We train the same classifier mentioned in Safranchik et al. (2020) on the weakly labeled training data obtained by each algorithm on all benchmark datasets. We also keep the test data identical per each dataset and report the obtained performance on test data using micro-entity precision, recall, and F1 score. According to the results displayed in Table 2 (lower), the proposed DWS outperforms the next best model by an F1 score of 3.95% on CoNLL2003, which is arguably the hardest dataset given the analysis in Section 5.1. Additionally, DWS outperforms the closest competitor on NCBI-Disease by an F1 score of 0.83%, and achieves the second best performance on both BC5CDR and LaptopReview.

The purpose of reporting performance separately in two steps is to clearly demonstrate the improvement of performance obtained by weak supervision alone, which only compares the quality of learned annotations with true labels in Step 1. To our understandings, this is important because in practice, model selection and quality control of weak supervision should also focus on the quality of learned annotations. The second step, after the training data is automatically annotated, focuses on the bias-variance problem in a supervised scenario and the main goal is to select the best classifier to generalize well on unseen data.

5.3 Performance vs Entropy of Weak Labels

As previously discussed, comparing F1 scores alone across weak supervision approaches without knowing the difficulties of the problem could drive misleading conclusions. For example, if the weak labels provided for a dataset are highly correlated, it becomes a trivial problem to denoise them - meaning algorithms will have indistinguish-

Step 1

Model	CoNLL2003			BC5CDR			NCBI-Disease			LaptopReview		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Majority Vote	71.11	69.98	70.54	86.19	83.52	84.83	74.78	63.42	68.63	67.09	60.88	63.83
Snorkel	68.45	72.23	70.28	81.66	80.63	81.26	77.33	65.05	70.66	67.57	61.01	64.12
HMM1*	73.15	72.09	72.62	**89.72**	70.25	78.80	80.27	41.25	54.50	67.88	59.52	63.43
HMM2*	N/A	N/A	N/A	85.73	**83.96**	84.84	77.76	63.69	70.03	68.09	**61.46**	**64.61**
Fuzzy-LSTM-CRF**	74.92	74.02	74.46	88.17	83.38	**85.71**	**80.97**	64.99	**72.11**	**68.57**	60.57	64.34
DWS (ours)	**75.58**	**76.65**	**76.11**	87.47	83.84	85.61	79.78	**65.74**	72.08	68.06	60.45	64.03

Step 2

Model	CoNLL2003			BC5CDR			NCBI-Disease			LaptopReview		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Majority Vote	71.51	71.71	71.60	82.40	82.06	82.23	70.98	71.72	71.31	65.79	59.57	62.51
Unweighted Vote	68.00	69.67	68.82	83.0	81.86	82.43	77.32	73.59	75.39	66.61	58.90	62.51
Snorkel	72.68	71.70	72.18	80.23	**84.35**	82.24	71.10	**76.00**	73.41	64.09	**63.09**	**63.54**
HMM1*	70.78	70.28	70.53	81.27	73.90	77.39	78.63	51.00	61.82	66.29	56.21	60.80
HMM2*	NA	NA	NA	80.21	84.30	82.21	72.21	70.54	71.34	66.17	59.94	62.86
Fuzzy-LSTM-CRF**	72.32	71.05	71.68	83.36	82.70	**83.03**	78.67	70.77	74.50	**69.42**	57.77	63.05
DWS (ours)	**76.96**	**75.32**	**76.13**	**84.15**	81.88	82.99	78.60	74.01	**76.22**	69.08	58.75	63.44

Table 2: **Step 1:** Training set results of the training labels produced by the weak supervision models. **Step 2:** Test set results of discriminators trained on the label predictions of the weakly supervised NER algorithms. All of the results in both tables are the average scores over 5 runs of each model using different random seeds.
* HMM1 is from Lison et al. (2020) and HMM2 is the Hidden Markov Model from Safranchik et al. (2020) without 'linking' rules. Note that we attempted to train HMM2 on CoNLL2003 but ran out of memory when using 64GB of RAM.
** We use our own implementation for the Fuzzy-LSTM-CRF. The results differ from Shang et al. (2018) since different weak labelers are used.

able F1 scores. To more clearly differentiate the performance of algorithms, we can calculate how their performance scales with the difficulty of the denoising task. Intuitively, we expect that better algorithms should be more robust to challenging denoising tasks. To demonstrate this, we first bucket tokens in the weakly labeled CoNLL2003 training dataset based on the level of disagreement among their weak labels, which can be thought of as the difficulty of denoising them, and then plot the differences in average token F1 scores between DWS and the comparative models in each bucket. To quantify the 'disagreement' of the weak labelers on a token, we use the entropy of the distribution of their non-abstained votes. To gain further insight into the strengths and weaknesses of each model, we group entropy and F1 calculations by 3 different class types: **Token Position**, **Positioned Class**, **Unpositioned Class**. These are described in Section 5.1. The results are plotted in Figure 3.

Figure 3 shows the improvement in F1 score obtained by DWS over other comparing methods at several different levels of entropy. The left figure shows that as entropy increases, generally the improvement of performance on unpositioned classes also increases. The middle figure shows that our model does not have much of an advantage in resolving disagreements over the position each token has within an entity (ie B,I,L,U), and that Fuzzy-LSTM-CRF is actually significantly better at high entropy levels. Most importantly though, when both types of disagreement are combined in Figure 3 (right), the advantage of DWS becomes very significant as the gaps in performance between itself and others are steadily increasing functions.

5.4 Ablation Studies

To better understand the importance of each component in DWS, we study how the performance of DWS on CoNLL2003 is affected by removing some design functions in DWS. Specifics of the experiments are as follows:

No Penalty: Sets the penalty given to illegal class transitions (defined in Section 4.4) to 0. Without this illegal class transition penalty, validation and test F1 scores drop by 1.27% and 2.34% respectively.

No Penalty No CRF: In addition to removing the penalty for illegal tag transitions as explained above, we treat the true tags as conditionally independent by replacing the CRF in $P(Y|X)$ with a token-wise softmax. Such changes result in lowering the test F1 score by 8.77%. This experiment helps to highlight the importance of modeling sequential dependencies since without it, the performance is lower than even majority voting.

No Warm Start: Warm starting by initializing the hard EM procedure with majority votes is helpful

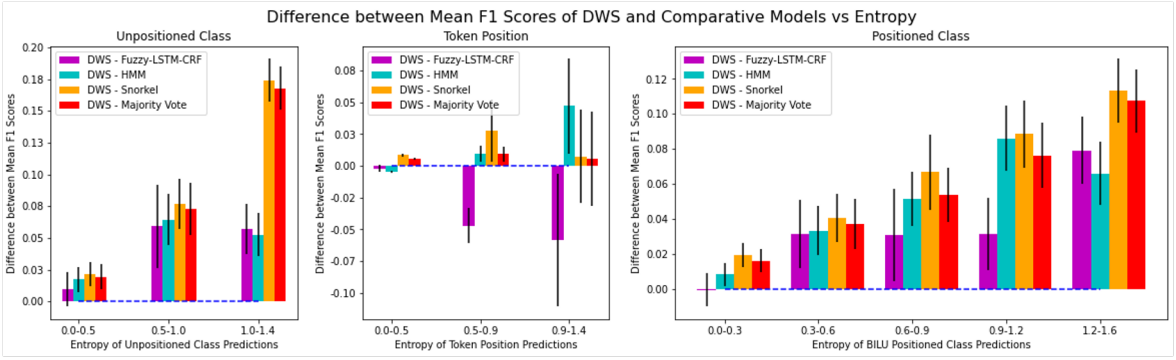

Figure 3: Each plot contains 95% confidence intervals on the difference in mean token F1 scores between DWS and comparing models in each bucket. These were calculated using the results from 5 randomized runs for each model. When the entropy is large, there is a lot of disagreement; when it's zero, the weak labelers all agree on the same class.

Model	CoNLL2003					
	Dev Precision	Dev Recall	Dev F1	Test Precision	Test Recall	Test F1
DWS	79.13	77.36	78.23	76.96	75.32	76.13
DWS (No Penalty)	77.92	76.02	76.96	74.35	73.25	73.79
DWS (No Penalty or Crf)	72.88	69.54	71.16	69.56	65.30	67.36
DWS (No Warm Start)	77.39	76.77	77.07	73.43	73.40	73.41

Table 3: Validation and test set results of a discriminator trained on the label predictions of each model listed in the table. The micro-entity precision, recall, and F1 scores are averaged over 5 random seeds.

to achieve better performance compared to random initialization. After iterating hard EM for the same epochs, warm start boosts validation and test F1 scores by 1.16% and 2.72% respectively, compared to random initialization. The results of each ablation are reported in Table 3.

6 Discussion

In Section 5.1 we showed that the amount of disagreement between the weak labelers varied substantially between datasets. This insight is important when interpreting the effectiveness of a weak supervision model because on datasets containing very little disagreement, we wouldn't expect to be able to learn anything much different than majority voting. As the number of contradictions between the weak labeler votes increases, for the most part so does the amount of information to help gauge the accuracies of the weak labelers and surpass majority voting. This means that we would roughly expect a good algorithm to have its greatest performance advantage on CoNLL2003 followed by NCBI-Disease and BC5CDR, and lastly LaptopReview which did not contain many weak labeler disagreements. The results in Section 5.2 show that as we would hope, our method has the largest performance advantage on CoNLL2003 and retains strong results on the remaining datasets which have fewer disagreements.

In Section 5.3 ('Performance vs Entropy'), we more concretely show that the F1 score advantage of DWS over the other models when using positioned labels increases with respect to the amount of contradiction information. These advantages may be attributable to the learned proficiency representations of weak labelers that help discriminate noisy labels provided by labeling functions with low proficiencies.

Experiments in Section 5.3 also show that DWS is very effective at resolving disagreements on unpositioned classes but is more mediocre at denoising disagreements on the entity positioning. These results suggest that the ideal model to use in a practical scenario likely depends on the amount of each type of disagreement in the given weakly labeled dataset.

7 Conclusion

In this paper we introduced a novel method (DWS) for weakly supervised NER which learns context-dependent proficiency for labeling sources while also modeling sequential dependencies among weak labels, inputs, and true labels. The proposed approach integrates representation learning and graphical models within the weak supervision setting, and obtains better performance on public benchmark datasets than recently introduced approaches. The proposed method is quite generic

and can be applied to other sequential learning tasks in NLP or other modalities such as image analysis/computer vision by adopting various pre-trained domain-specific models to embed the input sequence.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1638–1649. Association for Computational Linguistics.

Stephen H. Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, Cassandra Xia, Souvik Sen, Alex Ratner, Braden Hancock, Houman Alborzi, Rahul Kuchhal, Chris Ré, and Rob Malkin. 2019. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, page 362–375, New York, NY, USA. Association for Computing Machinery.

Yoshua Bengio and Paolo Frasconi. 1996. Input-output hmms for sequence processing. *IEEE Trans. Neural Networks*, 7(5):1231–1249.

Yoshua Bengio, Renato De Mori, Giovanni Flammia, and Ralf Kompe. 1992. Global optimization of a neural network-hidden markov model hybrid. *IEEE Trans. Neural Networks*, 3(2):252–259.

Hervé Bourlard and Christian Wellekens. 1990. Links between markov models and multilayer perceptrons. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(12):1167–1178.

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.

Daniel Y. Fu, Mayee F. Chen, Frederic Sala, Sarah M. Hooper, Kayvon Fatahalian, and Christopher Ré. 2020. Fast and three-rious: Speeding up weak supervision with triplet methods. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3280–3291. PMLR.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. 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 4238–4248, Hong Kong, China. Association for Computational Linguistics.

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

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

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Yan Li and Heung-Yeung Shum. 2006. Learning dynamic audio-visual mapping with input-output hidden markov models. *IEEE Trans. Multim.*, 8(3):542–549.

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*, page 1054–1064. Association for Computing Machinery, New York, NY, USA.

Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1518–1533, Online. Association for Computational Linguistics.

Liyuan Liu, Xiang Ren, Qi Zhu, Shi Zhi, Huan Gui, Heng Ji, and Jiawei Han. 2017. Heterogeneous supervision for relation extraction: A representation learning approach. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 46–56. Association for Computational Linguistics.

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.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering.

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 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2019. Training complex models with multi-task weak supervision. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:4763–4771.

Esteban Safranchik, Shiying Luo, and Stephen Bach. 2020. Weakly supervised sequence tagging from noisy rules. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:5570–5578.

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.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Ré. 2019a. Learning dependency structures for weak supervision models. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6418–6427. PMLR.

Paroma Varma, Frederic Sala, Shiori Sagawa, Jason Alan Fries, Daniel Y. Fu, Saelig Khattar, Ashwini Ramamoorthy, Ke Xiao, Kayvon Fatahalian, James Priest, and Christopher Ré. 2019b. Multi-resolution weak supervision for sequential data. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 192–203.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

A Appendix

A.1 Derivations

The details for efficiently computing Y' in step 1 of the hard EM algorithm as mentioned in Section 4.4 are as follows:

$$Y' = \arg\max_{Y_c \in \mathcal{E}^N} \Big(\log(P(\mathcal{L}, Y_c|\mathcal{X})) \Big) \tag{15}$$

$$= \arg\max_{Y_c \in \mathcal{E}^N} \Big(\log(P(Y_c|\mathcal{X})) + \sum_{i=1}^{N}\sum_{j=1}^{m} \log(P(l_{i,j}|\mathcal{X}, y_{c,i})) \Big) \tag{16}$$

$$= \arg\max_{Y_c \in \mathcal{E}^N} \Big(\log(\frac{e^{C(X,Y_c)}}{K}) + \sum_{i=1}^{N}\sum_{j=1}^{m} \log\left(P(l_{i,j}|\mathcal{X}, y_{c,i})\right) \Big) \tag{17}$$

$$= \arg\max_{Y_c \in \mathcal{E}^N} \Big(\sum_{i=1}^{N} s_{i,y_{c,i}} + \sum_{k=1}^{N-1} T[y_{c,k}, y_{c,k+1}] + \sum_{i=1}^{N}\sum_{j=1}^{m} \log(P(l_{i,j}|X, y_{c,i})) \Big)$$

$$= \arg\max_{Y_c \in \mathcal{E}^N} \Big(\sum_{i=1}^{N} s'_{i,y_{c,i}} + \sum_{k=1}^{N-1} T[y_{c,k}, y_{c,k+1}] \Big) \tag{18}$$

$$\text{where} \quad s'_{i,y_{c,i}} = s_{i,y_{c,i}} + \sum_{j=1}^{m} \log\left(P(l_{i,j}|X, y_{c,i})\right)$$

where K is a normalizing constant and C is the function defined in equation 10.

A.2 Graphical Model of DWS

We illustrate the exact graphical model of DWS in Figure 4, where the structure is similar to the Input/Output HMM(IOHMM) proposed by (Bengio and Frasconi, 1996). In IOHMM, transitions from latent variables y_{i-1} to y_i are directional, and analytical solutions are available based on factorizations of the sequence marginal likelihood. Here in our approach, similar analytical solutions could be derived as a partially directed model (e.g., CRF) without the need of parameterized distributions of input embeddings z_i. However, hybrid approaches that integrate neural networks with graphical models to enable efficient and scalable model training are preferred (Bourlard and Wellekens, 1990; Bengio et al., 1992; Bengio and Frasconi, 1996; Li and Shum, 2006).

A.3 Weak Supervision Model Architecture / Training Details

Weak Supervision Model Architectures: DWS and the Fuzzy-LSTM-CRF used a one layer

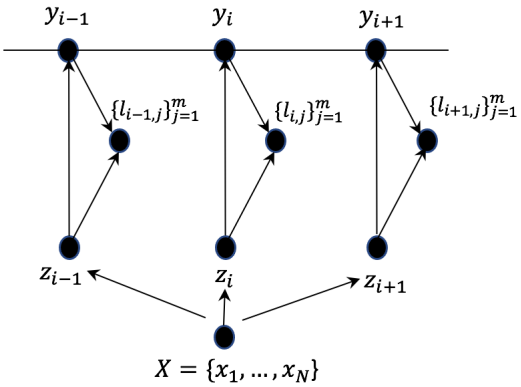

Figure 4: The input sequence $X = \{x_1, ..., x_N\}$ is transformed to a series of contextual embeddings $\{z_1, ..., z_N\}$ using a pre-trained BERT model followed by a BiLSTM. The conditional dependencies between input token embeddings z_i, latent NE tags y_i, and weak labels $\{l_{i,j}{}_{j=1}^{m}\}$ are modeled as functions of linear products between labeler representation θ_j and token embeddings z_i as formulated in equation 8. The dependencies between latent tags y_{i-1}, y_i are represented as an undirected graph through a CRF.

bidirectional LSTM with hidden size 768 to obtain the contextual token embeddings with dimension 768. The inputs to the BiLSTM were contextual word embeddings from RoBERTa (Liu et al., 2019) on CoNLL2003, uncased BERT on Laptop Review, and BioBERT (Lee et al., 2019) on NCBI-Disease and BC5CDR. These variants of BERT (Devlin et al., 2019) were not fine-tuned during training. Additionally, the class embeddings have dimension 768.

Training Details DWS: The optimizer used was RMSProp with learning rate 0.001. The model parameters were chosen to be those that gave the best micro-entity F1 score on the validation set over 10 random restarts of 5 epochs each. The batch sizes used were 32 for LaptopReview, 128 for NCBI-Disease, 256 for CoNLL2003, and 512 for BC5CDR. We defined the batch size by the number of tokens with at least 1 weak vote. Additionally, the illegal transition penalty was -2 for LaptopReview and -10 for the remaining datasets. Lastly, we used 1 warm-up epoch on majority vote labels and a dropout probability of 0.1 in the BiLSTM. These hyperparameters were chosen through manual tuning on the validation set.

Training Details Fuzzy-LSTM-CRF: The optimizer used was RMSProp with learning rate 0.01. The model parameters were chosen to be those that gave the best micro-entity F1 score on the validation set over 5 random restarts of 10 epochs each since it converged more slowly than DWS and had less variance among the random restarts. The best batch sizes were found to be the same as DWS. Lastly, the dropout probability in the BiLSTM was 0.1. The learning rate was chosen from $\{0.0001, 0.0005, 0.001, 0.01, 0.02\}$, the batch size was chosen from $\{32, 64, 128, 256, 512, 1024\}$, and the dropout was chosen from $\{0, 0.1, 0.2, 0.4\}$.

Training Details of Other Comparing Models: We used the same priors for the HMM1 model as were used in Lison et al. (2020) on CoNLL2003. For the remaining datasets the priors were tuned the on the validation sets. Additionally, we tried running the HMM from Safranchik et al. (2020) on CoNLL2003 but ran out of memory when using 64GB of RAM even when reducing the batch size to 1 and using unpositioned classes rather than BILU positioned classes (reduces number of classes by a factor of 4).

To improve the performance of *majority vote* and *unweighted vote* on CoNLL2003, which has many very noisy labels, we predicted 'No Entity' with probability 1 when there were less than T votes which was a tuned hyperparameter. The best value of T was found to be 5 for both methods and was chosen from $\{1, 5, 10\}$.

A.4 Discriminator Model Architecture / Training Details

The architecture of the discriminator was the same as used in Safranchik et al. (2020) which consisted of a 2 layer BiLSTM with 200 dimensional embeddings along with a CRF as the output layer. The model used both word embeddings from a variant of BERT and character embeddings from a CNN

as input. Additionally, we used their noise aware loss function.