BERTifying the Hidden Markov Model for Multi-Source Weakly Supervised Named Entity Recognition

Yinghao Li[1], **Pranav Shetty**[1], **Lucas Liu**[1], **Chao Zhang**[1], and **Le Song**[2]
[1] Georgia Institute of Technology, Atlanta, USA
[2] Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates
{yinghaoli, pranav.shetty, lucasliu, chaozhang}@gatech.edu
le.song@mbzuai.ac.ae

Abstract

We study the problem of learning a named entity recognition (NER) tagger using noisy labels from multiple weak supervision sources. Though cheap to obtain, the labels from weak supervision sources are often incomplete, inaccurate, and contradictory, making it difficult to learn an accurate NER model. To address this challenge, we propose a conditional hidden Markov model (CHMM), which can effectively infer true labels from multi-source noisy labels in an unsupervised way. CHMM enhances the classic hidden Markov model with the contextual representation power of pre-trained language models. Specifically, CHMM learns token-wise transition and emission probabilities from the BERT embeddings of the input tokens to infer the latent true labels from noisy observations. We further refine CHMM with an alternate-training approach (CHMM-ALT). It fine-tunes a BERT-NER model with the labels inferred by CHMM, and this BERT-NER's output is regarded as an additional weak source to train the CHMM in return. Experiments on four NER benchmarks from various domains show that our method outperforms state-of-the-art weakly supervised NER models by wide margins.

1 Introduction

Named entity recognition (NER), which aims to identify named entities from unstructured text, is an information extraction task fundamental to many downstream applications such as event detection (Li et al., 2012), relationship extraction (Bach and Badaskar, 2007), and question answering (Khalid et al., 2008). Existing NER models are typically supervised by a large number of training sequences, each pre-annotated with token-level labels. In practice, however, obtaining such labels could be prohibitively expensive. On the other hand, many domains have various knowledge resources such as knowledge bases, domain-specific dictionaries, or labeling rules provided by domain experts (Farmakiotou et al., 2000; Nadeau and Sekine, 2007). These resources can be used to match a corpus and quickly create large-scale noisy training data for NER from multiple views.

Learning an NER model from multiple weak supervision sources is a challenging problem. While there are works on distantly supervised NER that use only knowledge bases as weak supervision (Mintz et al., 2009; Shang et al., 2018; Cao et al., 2019; Liang et al., 2020), they cannot leverage complementary information from multiple annotation sources. To handle multi-source weak supervision, several recent works (Nguyen et al., 2017; Safranchik et al., 2020; Lison et al., 2020) leverage the hidden Markov model (HMM), by modeling true labels as hidden variables and inferring them from the observed noisy labels through unsupervised learning. Though principled, these models fall short in capturing token semantics and context information, as they either model input tokens as one-hot observations (Nguyen et al., 2017) or do not model them at all (Safranchik et al., 2020; Lison et al., 2020). Moreover, the flexibility of HMM is limited as its transitions and emissions remain constant over time steps, whereas in practice they should depend on the input words.

We propose the conditional hidden Markov model (CHMM) to infer true NER labels from multi-source weak annotations. CHMM conditions the HMM training and inference on BERT by predicting token-wise transition and emission probabilities from the BERT embeddings. These token-wise probabilities are more flexible than HMM's constant counterpart in modeling how the true labels should evolve according to the input tokens. The context representation ability they inherit from BERT also relieves the Markov constraint and expands HMM's context-awareness.

Further, we integrate CHMM with a supervised BERT-based NER mode with an alternate-training method (CHMM-ALT). It fine-tunes BERT-NER with the denoised labels generated by CHMM. Taking advantage of the pre-trained knowledge contained in BERT, this process aims to refine the denoised labels by discovering the entity patterns neglected by all of the weak sources. The fine-tuned BERT-NER serves as an additional supervision source, whose output is combined with other weak labels for the next round of CHMM training. CHMM-ALT trains CHMM and BERT-NER alternately until the result is optimized.

Our contributions include:

- A multi-source label aggregator CHMM with token-wise transition and emission probabilities for aggregating multiple sets of NER labels from different weak labeling sources.
- An alternate-training method CHMM-ALT that trains CHMM and BERT-NER in turn utilizing each other's outputs for multiple loops to optimize the multi-source weakly supervised NER performance.
- A comprehensive evaluation on four NER benchmarks from different domains demonstrates that CHMM-ALT achieves a 4.83 average F1 score improvement over the strongest baseline models.

The code and data used in this work are available at github.com/Yinghao-Li/CHMM-ALT.

2 Related Work

Weakly Supervised NER There have been works that train NER models with different weak supervision approaches. *Distant supervision*, a specific type of weak supervision, generates training labels from knowledge bases (Mintz et al., 2009; Yang et al., 2018; Shang et al., 2018; Cao et al., 2019; Liang et al., 2020). But such a method is limited to one source and falls short of acquiring supplementary annotations from other available resources. Other works adopt multiple additional labeling sources, such as heuristic functions that depend on lexical features, word patterns, or document information (Nadeau and Sekine, 2007; Ratner et al., 2016), and unify their results through multi-source *label denoising*. Several multi-source weakly supervised learning approaches are designed for sentence classification (Ratner et al., 2017, 2019; Ren et al., 2020; Yu et al., 2020). Although these methods can be adapted for sequence labeling tasks such as NER, they tend to overlook the internal dependency relationship between token-level labels during the inference. Fries et al. (2017) target the NER task, but their method first generates candidate named entity spans and then classifies each span independently. This independence makes it suffer from the same drawback as sentence classification models.

A few works consider label dependency while dealing with multiple supervision sources. Lan et al. (2020) train a BiLSTM-CRF network (Huang et al., 2015) with multiple parallel CRF layers, each for an individual labeling source, and aggregate their transitions with confidence scores predicted by an attention network (Bahdanau et al., 2015; Luong et al., 2015). HMM is a more principled model for multi-source sequential label denoising as the true labels are implicitly inferred through unsupervised learning without deliberately assigning any additional scores. Following this track, Nguyen et al. (2017) and Lison et al. (2020) use a standard HMM with multiple observed variables, each from one labeling source. Safranchik et al. (2020) propose linked HMM, which differs from ordinary HMM by introducing unique linking rules as an adjunct supervision source additional to general token labels. However, these methods fail to utilize the context information embedded in the tokens as effectively as CHMM, and their NER performance is further constrained by the Markov assumption.

Neuralizing the Hidden Markov Model Some works attempt to neuralize HMM in order to relax the Markov assumption while maintaining its generative property (Kim et al., 2018). For example, Dai et al. (2017) and Liu et al. (2018) incorporate recurrent units into the hidden semi-Markov model (HSMM) to segment and label high-dimensional time series; Wiseman et al. (2018) learn discrete template structures for conditional text generation using neuralized HSMM. Wessels and Omlin (2000) and Chiu and Rush (2020) factorize HMM with neural networks to scale it and improve its sequence modeling capacity. The work most related to ours leverages neural HMM for sequence labeling (Tran et al., 2016). CHMM differs from neural HMM in that the tokens are treated as a dependency term in CHMM instead of the observation in neural HMM. Besides, CHMM is trained with generalized EM, whereas neural HMM opti-

	Rockefeller	Center	in	New	York	was...
Source 1	B-PER	O	O	B-LOC	I-LOC	O...
Source 2	B-LOC	I-LOC	O	O	B-LOC	O...
Target	B-LOC	I-LOC	O	B-LOC	I-LOC	O...

Figure 1: An example of label aggregation with two weak labeling sources. We use BIO labeling scheme. `PER` represents person; `LOC` is location.

mizes the marginal likelihood of the observations.

3 Problem Setup

In this section, we formulate the multi-source weakly supervised NER problem. Consider an input sentence that contains T tokens $\boldsymbol{w}^{(1:T)}$, NER can be formulated as a sequence labeling task that assigns a label to each token in the sentence.[1] Assuming the set of target entity types is $\mathcal{E}$ and the tagging scheme is BIO (Ramshaw and Marcus, 1995), NER models assign one label from the label set $l \in \mathcal{L}$ to each token, where the size of the label set is $|\mathcal{L}| = 2|\mathcal{E}| + 1$, *e.g.*, if $\mathcal{E} = \{\texttt{PER}, \texttt{LOC}\}$, then $\mathcal{L} = \{\texttt{O}, \texttt{B-PER}, \texttt{I-PER}, \texttt{B-LOC}, \texttt{I-LOC}\}$.

Suppose we have a sequence with K weak sources, each of which can be a heuristic rule, knowledge base, or existing out-of-domain NER model. Each source serves as a labeling function that generates token-level weak labels from the input corpus, as shown in Figure 1. For the input sequence $\boldsymbol{w}^{(1:T)}$, we use $\boldsymbol{x}_k^{(1:T)}, k \in \{1, \dots, K\}$ to represent the weak labels from the source k, where $\boldsymbol{x}_k^{(t)} \in \mathbb{R}^{|\mathcal{L}|}, t \in \{1, \dots, T\}$ is a probability distribution over $\mathcal{L}$. Multi-source weakly supervised NER aims to find the underlying true sequence of labels $\hat{\boldsymbol{y}}^{(1:T)}, \hat{y}^{(t)} \in \mathcal{L}$ given $\{\boldsymbol{w}^{(1:T)}, \boldsymbol{x}_{1:K}^{(1:T)}\}$.

4 Methodology

In this section, we describe our proposed method CHMM-ALT. We first sketch the alternate-training procedure (§ 4.1), then explain the CHMM component (§ 4.2) and how BERT-NER is involved (§ 4.3).

4.1 Alternate-Training Procedure

The alternate-training method trains two models—a multi-source label aggregator CHMM and a BERT-NER model—in turn with each other's output. CHMM aggregates multiple sets of labels from different sources into a unified sequence of labels, while BERT-NER refines them by its language modeling ability gained from pre-training. The training process is divided into two phases.

- In **phase I**, CHMM takes the annotations $\boldsymbol{x}_{1:K}^{(1:T)}$ from existing sources and gives a set of denoised labels $\boldsymbol{y}^{*(1:T)}$, which are used to fine-tune the BERT-NER model. Then, we regard the fine-tuned model as an additional labeling source, whose outputs $\tilde{\boldsymbol{y}}^{(1:T)}$ are added into the original weak label sets to give the updated observation instances: $\boldsymbol{x}_{1:K+1}^{(1:T)} = \{\boldsymbol{x}_{1:K}^{(1:T)}, \tilde{\boldsymbol{y}}^{(1:T)}\}$.
- In **phase II**, CHMM and BERT-NER mutually improve each other iteratively in several loops. Each loop first trains CHMM with the observation $\boldsymbol{x}_{1:K+1}^{(1:T)}$ from the previous one. Then, its predictions are adopted to fine-tune BERT-NER, whose output updates $\boldsymbol{x}_{K+1}^{(1:T)}$.

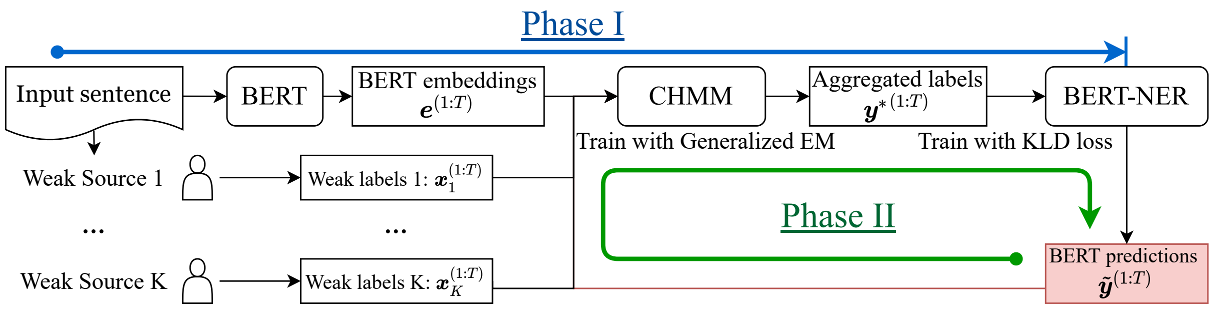
Figure 2 illustrates the alternate-training method. In general, CHMM gives high precision predictions, whereas BERT-NER trades recall with precision. In other words, CHMM can classify named entities with high accuracy but is slightly disadvantaged in discovering all entities. BERT-NER increases the coverage with a certain loss of accuracy. Combined with the alternate-training approach, this complementarity between these models further increases the overall performance.

4.2 Conditional Hidden Markov Model

The conditional hidden Markov model is an HMM variant for multi-source label denoising. It models true entity labels as hidden variables and infers them from the observed noisy labels. Traditionally, discrete HMM uses *one* transition matrix to model the probability of hidden label transitioning and *one* emission matrix to model the probability of the observations from the hidden labels. These two matrices are constant, *i.e.*, their values do not change over time steps. CHMM, on the contrary, conditions both its transition and emission matrices on the BERT embeddings $\boldsymbol{e}^{(1:T)}$ of the input tokens $\boldsymbol{w}^{(1:T)}$. This design not only allows CHMM to leverage the rich contextual representations of the BERT embeddings but relieves the constant matrices constraint as well.

In phase I, CHMM takes K sets of weak labels from the provided K weak labeling sources. In phase II, in addition to the existing sources, it takes

[1] We represent vectors, matrices or tensors with bold fonts and scalars with regular fonts; $1 : a \triangleq \{1, 2, \dots, a\}$.



Phase I
Input sentence
BERT
BERT embeddings $e^{(1:T)}$
CHMM
Aggregated labels $y^{*(1:T)}$
BERT-NER
Train with Generalized EM
Train with KLD loss
Weak Source 1
Weak labels 1: $x_1^{(1:T)}$
...
Phase II
Weak Source K
Weak labels K: $x_K^{(1:T)}$
BERT predictions $\tilde{y}^{(1:T)}$

Figure 2: The illustration of the alternate-training method. Phase I is acyclic, starting from getting K weak labels from supervision sources and ending at the fine-tuning of BERT-NER with CHMM's denoised output. Phase II contains several loops, each trains CHMM with $K+1$ sources, including the additional BERT predictions from the previous loop, and fine-tunes BERT-NER using the updated denoised labels.

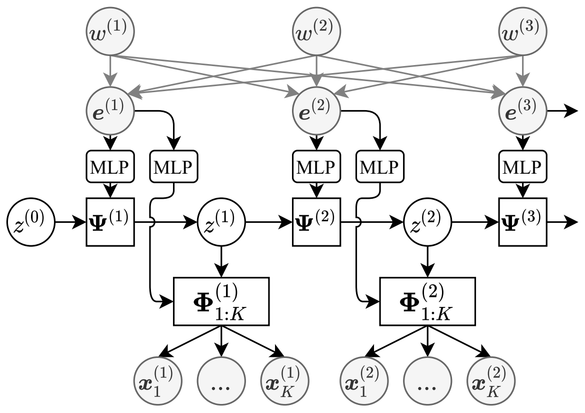


Figure 3: An illustration of CHMM's architecture. Shaded circles are observed elements; white circles are hidden elements; rectangles are matrices. Rounded rectangles are multi-layer perceptrons containing the trainable parameters. The arrows between $\boldsymbol{w}^{(t)}$ and $\boldsymbol{e}^{(t)}$ denote the context representation ability of BERT. MLP denotes the "multi-layer perceptron".

another set of labels from the previously fine-tuned BERT-NER, making the total number of sources $K+1$. For convenience, we use K as the number of weak sources below.

Model Architecture Figure 3 shows a sketch of CHMM's architecture.[2] $\boldsymbol{z}^{(1:T)}$ denotes the discrete hidden states of CHMM with $z^{(t)} \in \mathcal{L}$, representing the underlying true labels to be inferred from multiple weak annotations. $\boldsymbol{\Psi}^{(t)} \in \mathbb{R}^{|\mathcal{L}|\times|\mathcal{L}|}$ is the transition matrix, whose element $\Psi_{i,j}^{(t)} = p(z^{(t)} = j|z^{(t-1)} = i, \boldsymbol{e}^{(t)}), i,j \in \{1,\dots,|\mathcal{L}|\}$ denotes the probability of moving from label i to label j at time step t. $\boldsymbol{\Phi}_k^{(t)} \in \mathbb{R}^{|\mathcal{L}|\times|\mathcal{L}|}$ is the emission matrix of weak source k, each element in which $\Phi_{i,j,k}^{(t)} = p(x_{j,k}^{(t)} = 1|z^{(t)} = i, \boldsymbol{e}^{(t)})$ represents the probability of source k observing label j when the hidden label is i at time step t.

[2]We relax plate notation here to present details.

For each step, $\boldsymbol{e}^{(t)} \in \mathbb{R}^{d_{\text{emb}}}$ is the output of a pre-trained BERT with d_{emb} being its embedding dimension. $\boldsymbol{\Psi}^{(t)}$ and $\boldsymbol{\Phi}_{1:K}^{(t)}$ are calculated by applying a multi-layer perceptron (MLP) to $\boldsymbol{e}^{(t)}$:

$$\boldsymbol{s}^{(t)} \in \mathbb{R}^{|\mathcal{L}|^2} = \text{MLP}(\boldsymbol{e}^{(t)}), \tag{1}$$

$$\boldsymbol{h}^{(t)} \in \mathbb{R}^{|\mathcal{L}|\cdot|\mathcal{L}|\cdot K} = \text{MLP}(\boldsymbol{e}^{(t)}). \tag{2}$$

Since the MLP outputs are vectors, we need to reshape them to matrices or tensors:

$$\boldsymbol{S}^{(t)} \in \mathbb{R}^{|\mathcal{L}|\times|\mathcal{L}|} = \text{reshape}(\boldsymbol{s}^{(t)}), \tag{3}$$

$$\boldsymbol{H}^{(t)} \in \mathbb{R}^{|\mathcal{L}|\times|\mathcal{L}|\times K} = \text{reshape}(\boldsymbol{h}^{(t)}). \tag{4}$$

To achieve the proper probability distributions, we apply the Softmax function along the *label* axis so that these values are positive and sum up to 1:

$$\boldsymbol{\Psi}_{i,1:|\mathcal{L}|}^{(t)} = \sigma(\boldsymbol{S}_{i,1:|\mathcal{L}|}^{(t)}),\ \boldsymbol{\Phi}_{i,1:|\mathcal{L}|,k}^{(t)} = \sigma(\boldsymbol{H}_{i,1:|\mathcal{L}|,k}^{(t)}),$$

where

$$\sigma(\boldsymbol{a})_i = \frac{\exp(a_i)}{\sum_j \exp(a_j)}. \tag{5}$$

$\boldsymbol{a}$ is an arbitrary vector. The formulae in the following discussion always depend on $\boldsymbol{e}^{(1:T)}$, but we will omit the dependency term for simplicity.

Model Training According to the generative process of CHMM, the joint distribution of the hidden states and the observed weak labels for one sequence $p(\boldsymbol{z}^{(0:T)}, \boldsymbol{x}^{(1:T)}|\boldsymbol{\theta})$ can be factorized as:

$$\begin{aligned} p(\boldsymbol{z}^{(0:T)}, \boldsymbol{x}^{(1:T)}|\boldsymbol{\theta}) &= p(z^{(0)})p(\boldsymbol{x}^{(1:T)}|\boldsymbol{z}^{(1:T)}) \\ &= p(z^{(0)})\prod_{t=1}^{T} p(z^{(t)}|z^{(t-1)})\prod_{t=1}^{T} p(\boldsymbol{x}^{(t)}|z^{(t)}), \end{aligned} \tag{6}$$

where $\boldsymbol{\theta}$ represents all the trainable parameters.

HMM is generally trained with an expectation-maximization (EM, also known as Baum-Welch) algorithm. In the expectation step (E-step), we compute the expected complete data log likelihood:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) \triangleq \mathbb{E}_{\boldsymbol{z}}[\ell_c(\boldsymbol{\theta}) | \boldsymbol{\theta}^{\text{old}}]. \tag{7}$$

$\boldsymbol{\theta}^{\text{old}}$ is the parameters from the previous training step, $\mathbb{E}_{\boldsymbol{z}}[\cdot]$ is the expectation over variable $\boldsymbol{z}$, and

$$\ell_c(\boldsymbol{\theta}) \triangleq \log p(\boldsymbol{z}^{(0:T)}, \boldsymbol{x}^{(1:T)} | \boldsymbol{\theta})$$

is the comptelete data log likelihood. Let $\boldsymbol{\varphi}^{(t)} \in \mathbb{R}^{|\mathcal{L}|}$ be the observation likelihood where

$$\varphi_i^{(t)} \triangleq p(\boldsymbol{x}^{(t)} | z^{(t)} = i) = \prod_{k=1}^{K} \sum_{j=1}^{|\mathcal{L}|} \Phi_{i,j,k}^{(t)} x_{j,k}^{(t)}. \tag{8}$$

Combining (6)–(8) together, we have

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{i=1}^{|\mathcal{L}|} \gamma_i^{(0)} \log \pi_i + \sum_{t=1}^{T} \sum_{i=1}^{|\mathcal{L}|} \sum_{j=1}^{|\mathcal{L}|} \xi_{i,j}^{(t)} \log \Psi_{i,j}^{(t)} + \sum_{t=1}^{T} \sum_{i=1}^{|\mathcal{L}|} \gamma_i^{(t)} \log \varphi_i^{(t)}, \tag{9}$$

where $\pi_1 = 1, \boldsymbol{\pi}_{2:|\mathcal{L}|} = \boldsymbol{0}$;[3] $\gamma_i^{(t)} \triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:T)})$ is the smoothed marginal; $\xi_{i,j}^{(t)} \triangleq p(z^{(t-1)} = i, z^{(t)} = j | \boldsymbol{x}^{(1:T)})$ is the expected number of transitions. These parameters are computed using the *forward-backward* algorithm.[4]

In the maximization step (M-step), traditional HMM updates parameters $\boldsymbol{\theta}_{\text{HMM}} = \{\boldsymbol{\Psi}, \boldsymbol{\Phi}, \boldsymbol{\pi}\}$ by optimizing (7) with pseudo-statistics.[5] However, as the transitions and emissions in CHMM are not standalone parameters, we cannot directly optimize CHMM by this method. Instead, we update the model parameters through gradient descent *w.r.t.* $\boldsymbol{\theta}_{\text{CHMM}}$ using (9) as the objective function:

$$\nabla \boldsymbol{\theta}_{\text{CHMM}} = \frac{\partial Q(\boldsymbol{\theta}_{\text{CHMM}}, \boldsymbol{\theta}_{\text{CHMM}}^{\text{old}})}{\partial \boldsymbol{\theta}_{\text{CHMM}}}. \tag{10}$$

In practice, the calculation is conducted in the logarithm domain to avoid the loss of precision issue that occurs when the floating-point numbers become too small.

To solve the label sparsity issue, *i.e.*, some entities are only observed by a minority of the weak sources, we modify the observations $\boldsymbol{x}^{(1:T)}$ before training. If one source k observes an entity at time step t: $x_{j\neq 1,k}^{(t)} > 0$, the observation of non-observing sources at t will be modified to $\boldsymbol{x}_{1,\kappa}^{(t)} = \epsilon; \boldsymbol{x}_{j\neq 1,\kappa}^{(t)} = (1-\epsilon)/|\mathcal{L}|, \forall \kappa \in \{1, \dots, K\} \backslash k$, where ϵ is an arbitrary small value. Note that $x_{1,\kappa}^{(t)}$ corresponds to the observed label O.

CHMM Initialization Generally, HMM has its transition and emission probabilities initialized with the statistics $\boldsymbol{\Psi}^*$ and $\boldsymbol{\Phi}^*$ computed from the observation set. But it is impossible to directly set $\boldsymbol{\Psi}^{(t)}$ and $\boldsymbol{\Phi}^{(t)}$ in CHMM to these values, as these matrices are the output of the MLPs rather than standalone parameters. To address this issue, we choose to pre-train the MLPs before starting CHMM's training by minimizing the mean squared error (MSE) loss between their outputs and the target statistics:

$$\ell_{\text{MSE}} = \frac{1}{T} \sum_t \|\boldsymbol{\Psi}^* - \boldsymbol{S}^{(t)}\|_F^2 + \|\boldsymbol{\Phi}^* - \boldsymbol{H}^{(t)}\|_F^2,$$

where $\|\cdot\|_F$ is the Frobenius norm. Right after initialization, MLPs can only output similar probabilities for all time steps: $\boldsymbol{\Psi}^{(t)} \approx \boldsymbol{\Psi}^*$, $\boldsymbol{\Phi}^{(t)} \approx \boldsymbol{\Phi}^*$, $\forall t \in \{1, 2, \dots, T\}$. But their token-wise prediction divergence will emerge when CHMM has been trained. The initial hidden state $z^{(0)}$ is fixed to O as it has no corresponding token.

Inference Once trained, CHMM can provide the most probable sequence of hidden labels $\hat{\boldsymbol{z}}^{(1:T)}$ along with the probabilities of all labels $\boldsymbol{y}^{*(1:T)}$.

$$\hat{\boldsymbol{z}}^{(1:T)} = \arg\max_{\boldsymbol{z}^{(1:T)}} p_{\hat{\boldsymbol{\theta}}_{\text{CHMM}}}(\boldsymbol{z}^{(1:T)} | \boldsymbol{x}_{1:K}^{(1:T)}, \boldsymbol{e}^{(1:T)}),$$
$$y_i^{*(t)} = p_{\hat{\boldsymbol{\theta}}_{\text{CHMM}}}(z^{(t)} = i | \boldsymbol{x}_{1:K}^{(1:T)}, \boldsymbol{e}^{(1:T)}),$$

where $\hat{\boldsymbol{\theta}}_{\text{CHMM}}$ represents the trained parameters. These results can be calculated by either the Viterbi decoding algorithm (Viterbi, 1967) or directly maximizing the smoothed marginal $\boldsymbol{\gamma}^{(1:T)}$.

4.3 Improving Denoised Labels with BERT

The pre-trained BERT model encodes semantic and structural knowledge, which can be distilled to further refine the denoised labels from CHMM. Specifically, we construct the BERT-NER model by stacking a feed-forward layer and a Softmax layer on top of the original BERT to predict the probabilities of the classes that each token belongs

[3] This assumes the initial hidden state is always O. In practice, we set $\pi_\ell = \epsilon, \forall \ell \in 2 : |\mathcal{L}|$ and $\pi_1 = 1 - (|\mathcal{L}| - 1)\epsilon$, where ϵ is a small value, to avoid getting $-\infty$ from log.

[4] Details are presented in appendix A.1.

[5] Details are presented in appendix A.2.

to (Sun et al., 2019). The probability predictions of CHMM, $\boldsymbol{y}^{*(1:T)}$, often referred to as *soft labels*, are chosen to supervise the fine-tuning procedure. Compared with the hard labels $\hat{\boldsymbol{z}}^{(1:T)}$, soft labels lead to a more stable training process and higher model robustness (Thiel, 2008; Liang et al., 2020).

We train BERT-NER by minimizing the Kullback-Leibler divergence (KL divergence) between the soft labels $\boldsymbol{y}^*$ and the model output $\boldsymbol{y}$:

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{\text{BERT}} &= \arg\min_{\boldsymbol{\theta}_{\text{BERT}}} \mathcal{D}[\boldsymbol{y}^{*(1:T)} \| \boldsymbol{y}^{(1:T)}] \\ &= \arg\min_{\boldsymbol{\theta}_{\text{BERT}}} \sum_{t=1}^{T} \sum_{i=1}^{|\mathcal{L}|} y^{*(t)}_i \log \frac{y^{*(t)}_i}{y^{(t)}_i},\end{aligned} \tag{11}$$

where $\boldsymbol{\theta}_{\text{BERT}}$ denotes all the trainable parameters in the BERT model. BERT-NER does not update the embeddings $\boldsymbol{e}^{(1:T)}$ that CHMM depends on.

We obtain the refined labels $\tilde{\boldsymbol{y}}^{(1:T)} \in \mathbb{R}^{T\times|\mathcal{L}|}$ from the fine-tuned BERT-NER directly through a forward pass. Different from CHMM, we continue BERT-NER's training with parameter weights from the last loop's checkpoint so that the model is initialized closer to the optimum. Correspondingly, phase II trains BERT-NER with a smaller learning rate, fewer epoch iterations, and batch gradient descent instead of the mini-batch version.[6] This strategy speeds up phase II training without sacrificing the model performance as $\boldsymbol{y}^{*(1:T)}$ does not change significantly from loop to loop.

5 Experiments

We benchmark CHMM-ALT on four datasets against state-of-the-art weakly supervised NER baselines, including both distant learning models and multi-source label aggregation models. We also conduct a series of ablation studies to evaluate the different components in CHMM-ALT's design.

5.1 Setup

Datasets We consider four NER datasets covering the general, technological and biomedical domains: 1) **CoNLL 2003** (English subset) (Tjong Kim Sang and De Meulder, 2003) is a general domain dataset containing 22,137 sentences manually labelled with 4 entity types. 2) **LaptopReview** dataset (Pontiki et al., 2014) consists of 3,845 sentences with laptop-related entity mentions. 3) **NCBI-Disease** dataset (Dogan et al., 2014) contains 793 PubMed abstracts annotated with disease mentions. 4) **BC5CDR** (Li et al., 2016), the dataset accompanies the BioCreative V CDR challenge, consists of 1,500 PubMed articles, annotated with chemical disease mentions.

[6]Hyper-parameter values are listed in appendix C.

	Co03	NCBI	CDR	LR
# Instance	22,137	793	1,500	3,845
# Training	14,041	593	500	2,436
# Development	3,250	100	500	609
# Test	3,453	100	500	800
Ave# Tokens	14.5	219.8	217.7	16.4
# Entities	4	1	2	1
# Sources	13	5	8	4

Table 1: Dataset statistics. Co03 is CoNLL 2003; LR is LaptopReview; CDR is BC5CDR. "# Sources" indicates the number of labeling sources for each dataset.

Table 1 shows dataset statistics, including the average number of tokens, entities and weak labeling sources. We use the original word tokens in the dataset if provided and use NLTK (Bird and Loper, 2004) otherwise for sentence tokenization.

For weak labeling sources, we use the ones from Lison et al. (2020) for CoNLL 2003, and the ones from Safranchik et al. (2020) for LaptopReview, NCBI-Disease and BC5CDR.[7]

Baselines We compare our model to the following state-of-the-art baselines: 1) **Majority Voting** returns the label for a token that has been observed by most of the sources and randomly chooses one if it's a tie; 2) **Snorkel** (Ratner et al., 2017) treats each token in a sequence as i.i.d. and conducts the label classification without considering its context; 3) **SwellShark** (Fries et al., 2017) improves Snorkel by predicting all the target entity spans before classifying them using naïve Bayes; 4) **AutoNER** (Shang et al., 2018) augments distant supervision by predicting whether two consecutive tokens should be in the same entity span; 5) **BOND** (Liang et al., 2020) adopts self-training and high-confidence selection to further boost the distant supervision performance. 6) **HMM** is the multi-observation generative model used in Lison et al. (2020) that does not have the integrated neural network; 7) **Linked HMM** (Safranchik et al., 2020) uses linking rules to provide additional inter-token structural information to the HMM model.

For the ablation study, we modify CHMM to another type of i.i.d. model by taking away its transition matrices. This model, named **CHMM-i.i.d.**,

[7]Details are presented in appendix B.

Models	CoNLL 2003	NCBI-Disease	BC5CDR	LaptopReview
Supervised BERT-NER ‡ ♮	90.74 (90.37/91.10)	88.89 (87.05/90.82)	88.81 (87.12/90.57)	81.34 (82.02/80.67)
best consensus ♮	89.18 (100.0/80.47)	81.60 (100.0/68.91)	87.58 (100.0/77.89)	77.72 (100.0/63.55)
SwellShark (noun-phrase) †‡	-	67.10 (64.70/69.70)	84.23 (84.98/83.49)	-
SwellShark (hand-tuned) †‡	-	80.80 (81.60/80.10)	84.21 (86.11/82.39)	-
AutoNER †‡	67.00 (75.21/60.40)	75.52 (79.42/71.98)	82.13 (83.23/81.06)	65.44 (72.27/59.79)
Snorkel †‡	66.40 (71.40/62.10)	73.41 (71.10/76.00)	82.24 (80.23/84.35)	63.54 (64.09/63.09)
Linked HMM †‡	-	79.03 (83.46/75.05)	82.96 (82.65/83.28)	69.04 (77.74/62.11)
BOND-MV †‡ ♮	65.96 (64.22/67.82)	80.33 (84.77/76.34)	83.18 (82.90/83.49)	67.19 (68.90/65.75)
Majority Voting † ♮	58.40 (49.01/72.24)	73.94 (79.76/68.91)	80.73 (83.79/77.88)	67.92 (72.93/63.55)
HMM † ♮	68.84 (70.80/66.98)	73.06 (83.88/64.70)	80.57 (88.75/73.76)	66.96 (77.46/58.96)
CHMM-i.i.d. † ♮	68.57 (69.67/67.50)	71.69 (83.49/62.87)	79.37 (85.68/73.92)	65.89 (75.70/58.34)
CHMM † ♮	70.11 (72.98/67.47)	78.88 (**93.37**/68.28)	82.39 (**89.93**/76.02)	73.02 (**87.23**/62.79)
CHMM + BERT-NER †‡ ♮	74.30 (75.02/73.58)	82.87 (89.42/77.22)	84.33 (85.58/83.12)	69.67 (75.48/64.70)
CHMM-ALT †‡ ♮	**75.54** (**76.22**/**74.86**)	**85.02** (87.92/**82.47**)	**85.12** (84.97/**85.28**)	**76.55** (81.39/**72.32**)

Table 2: Evaluation results on four datasets. The results are presented in the "F1 (Precision/Recall)" format. "CHMM + BERT-NER" is essentially CHMM-ALT's phase I output. "BOND-MV" is the BOND model trained with majority voted labels. † indicates unsupervised label denoiser; ‡ represents fully supervised models. A model with †‡ is either distantly supervised or trains a supervised by labels from the denoiser. ♮ signifies the results from our experiments. In addition to models with ♮, Snorkel and Linked HMM also share our labeling sources.

directly predicts the hidden steps from the BERT embeddings, while otherwise identical to CHMM. We also investigate how CHMM-ALT performs with other aggregators other than CHMM.

We also introduce two upper bounds from different aspects: 1) a **fully supervised BERT-NER** model trained with manually labeled data is regarded as a supervised reference; 2) the **best possible consensus** of the weak sources. The latter assumes an oracle that always selects the correct annotations from these weak supervision sources. According to the definition, its precision is always 100% and its recall is non-decreasing with the increase of the number of weak sources.

Evaluation Metrics We evaluate the performance of NER models using entity-level precision, recall, and F1 scores. All scores are presented as percentages. The results come from the average of 5 trials with different random seeds.

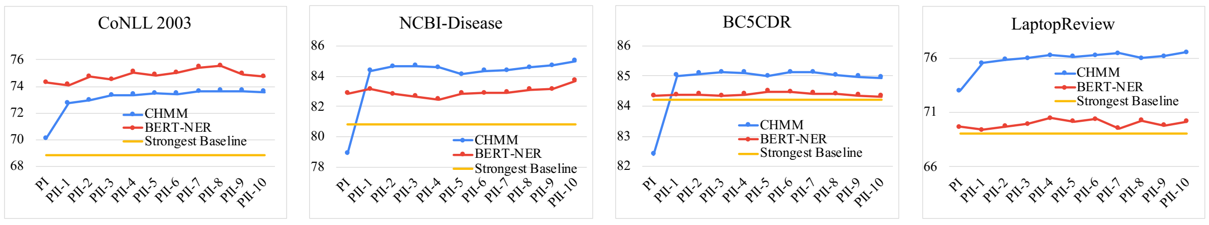
Implementation Details We use BERT pre-trained on different domains for different datasets, both for embedding construction and as the component of the supervised BERT-NER model. The original BERT (Devlin et al., 2019) is used for CoNLL 2003 and LaptopReview datasets, bioBERT (Lee et al., 2019) for NCBI-Disease and SciBERT (Beltagy et al., 2019) for BC5CDR. Instances with lengths exceeding BERT's maximum length limitation (512) are broken into several shorter segments.

The only tunable hyper-parameter in CHMM is the learning rate. But its influence is negligible—benefitted from the stability of the generalized EM, the model is guaranteed to converge to a local optimum if the learning rate is small enough. For all the BERT-NER models used in our experiments, the hyper-parameters except the batch size are fixed to the default values (appendix C).

To prevent overfitting, we use a two-scale early stopping strategy for model choosing at two scales based on the development set. The micro-scale early stopping chooses the best model parameters for each individual training process of both CHMM and BERT-NER; the macro-scale early stopping selects the best-performing model in phase II iterations, which reports the test results. In our experiments, phase II exits if the macro-scale development score has not increased in 5 loops or the maximum number of loops (10) is reached.

5.2 Main Results

Table 2 presents the model performance from different domains. We find that our alternate-training framework outperforms all weakly supervised baseline models. In addition, CHMM-ALT approaches or even exceeds the best source consensus, which sufficiently proves the effectiveness of the design. For general HMM-based label aggregators such as CHMM, it is impossible to exceed the best consensus since they can only predict an entity observed by at least one source. Based on this fact, CHMM is designed to select the most accurate observations from the weak sources without shrinking their coverage. In comparison, BERT's language



(a) (b) (c) (d)

Figure 4: F1 score evolution across the alternate-training phases. "PI" is phase I; "PII-i" is the ith loop of phase II. The "strongest baseline" reports the result from the best-performed baseline in Table 2 for each dataset.

representation ability enables it to generalize the entity patterns and successfully discovers those entities annotated by none of the sources. Comparing CHMM + BERT to CHMM, we can conclude that BERT basically exchanges recall with precision, and its high-recall predictions can improve the result of CHMM in return. The complementary nature of these two models is why CHMM-ALT improves the overall performance of weakly supervised NER.

5.3 Analysis of CHMM

Looking at Table 2, we notice that CHMM performs the best amongst all generative models including majority voting, HMM and CHMM-i.i.d. The performance of conventional HMM is largely limited by the Markov assumption with the unchanging transition and emission probabilities. The results in the table validate that conditioning the model on BERT embedding alleviates this limitation. However, the transition matrices in HMM are indispensable, implied by CHMM-i.i.d.'s results, as they provide supplemental information about how the underlying true labels should evolve.

5.4 Analysis of Alternate-Training

Performance Evolution Figure 4 reveals the details of the alternate-training process. For less ambiguous tasks including NCBI-Disease, BC5CDR and LaptopReview with fewer entity types, BERT generally has better performance in phase I but gets surpassed in phase II. Interestingly, BERT's performance never exceeds that of CHMM on the LaptopReview dataset. This may be because BERT fails to construct sufficiently representative patterns from the denoised labels for this dataset. For CoNLL 2003, where it is harder for the labeling sources to model the language structures, the strength of a pre-trained language model in pattern recognition becomes more prominent. From the results it seems that the performance increment of the denoised labels $\boldsymbol{y}^{*(1:T)}$ provides marginally extra information to BERT after phase II, as most of the increment comes from the information provided by BERT itself. Even so, keeping phase II is reasonable when we want to get the best out of the weak labeling sources and the pre-trained BERT.

Models	Co03	NCBI	CDR	Laptop
MV † ♮	58.40	73.94	80.73	67.92
MV-ALT †‡ ♮	66.64	80.83	82.78	70.45
HMM † ♮	68.84	73.06	80.57	66.96
HMM-ALT †‡ ♮	74.04	82.99	83.34	72.90
i.i.d. † ♮	68.57	71.69	79.37	65.89
i.i.d.-ALT †‡ ♮	73.84	83.15	83.17	72.61
CHMM † ♮	70.11	78.88	82.39	73.02
CHMM-ALT †‡ ♮	75.54	85.02	85.12	76.55

Table 3: Alternate-training F1 scores with different label aggregators. MV denotes Majority voting; i.i.d. represents CHMM-i.i.d. The model names without the "ALT" suffix are the multi-source label aggregators whereas the suffix indicates that the result comes from the alternate-training framework with the corresponding model as the label aggregator.

BERT-NER Initialization CHMM-ALT initializes BERT-NER's parameters from its previous checkpoint at the beginning of each loop in phase II to reduce training time (§ 4.3). If we instead fine-tune BERT-NER from the initial parameters of the pre-trained BERT model for each loop, CHMM-ALT gets 84.30, 84.71, and 76.68 F1 scores on NCBI-Disease, BC5CDR, and LaptopReview datasets. These scores are close to the results in Table 2, but the training takes much longer. Consequently, our BERT-NER initialization strategy is a more practical choice overall.

Applying Alternate-Training to Other Methods Table 3 shows the alternate-training performance acquired with different label aggregators. The accompanying BERT-NER models are identical to those described in § 5.1. The results in the table suggest that the performance improvement obtained by using alternate-training on the label aggregators is stable and generalizable to any other models yet to be proposed.

6 Conclusion

In this work, we present CHMM-ALT, a multi-source weakly supervised approach that does not depend on manually labeled data to learn an accurate NER tagger. It integrates a label aggregator—CHMM and a supervised model—BERT-NER together into an alternate-training procedure. CHMM conditions HMM on BERT embeddings to achieve greater flexibility and stronger context-awareness. Fine-tuned with CHMM's prediction, BERT-NER discovers patterns unobserved by the weak sources and complements CHMM. Training these models in turn, CHMM-ALT uses the knowledge encoded in both the weak sources and the pre-trained BERT model to improve the final NER performance. In the future, we will consider imposing more constraints on the transition and emission probabilities, or manipulating them according to sophisticated domain knowledge. This technique could be also extended to other sequence labeling tasks such as semantic role labeling or event extraction.

Acknowledgments

This work was supported by ONR MURI N00014-17-1-2656, NSF III-2008334, Kolon Industries, and research gifts from Google and Amazon. In addition, we would like to thank Yue Yu for his insightful suggestions for this work.

References

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Yixin Cao, Zikun Hu, Tat-seng Chua, Zhiyuan Liu, and Heng Ji. 2019. Low-resource name tagging learned with weakly labeled data. 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 261–270, Hong Kong, China. Association for Computational Linguistics.

Justin Chiu and Alexander Rush. 2020. Scaling hidden Markov language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1341–1349, Online. Association for Computational Linguistics.

Hanjun Dai, Bo Dai, Yan-Ming Zhang, Shuang Li, and Le Song. 2017. Recurrent hidden semi-markov model. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

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.

Rezarta Islamaj Dogan, Robert Leaman, and Zhiyong Lu. 2014. NCBI disease corpus: A resource for disease name recognition and concept normalization. *J. Biomed. Informatics*, 47:1–10.

Dimitra Farmakiotou, Vangelis Karkaletsis, John Koutsias, George Sigletos, Constantine D. Spyropoulos, and Panagiotis Stamatopoulos. 2000. Rule-based named entity recognition for greek financial texts. In *In Proceedings of the Workshop on Computational Lexicography and Multimedia Dictionaries (COMLEX 2000*, pages 75–78.

Jason A. Fries, Sen Wu, Alexander Ratner, and Christopher Ré. 2017. Swellshark: A generative model for biomedical named entity recognition without labeled data. *CoRR*, abs/1704.06360.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Mahboob Alam Khalid, Valentin Jijkoun, and Maarten De Rijke. 2008. The impact of named entity normalization on information retrieval for question answering. In *Proceedings of the IR Research, 30th European Conference on Advances in Information Retrieval*, ECIR'08, pages 705–710, Berlin, Heidelberg. Springer-Verlag.

Yoon Kim, Sam Wiseman, and Alexander M. Rush. 2018. A tutorial on deep latent variable models of natural language. *CoRR*, abs/1812.06834.

Ouyu Lan, Xiao Huang, Bill Yuchen Lin, He Jiang, Liyuan Liu, and Xiang Ren. 2020. Learning to contextually aggregate multi-source supervision for sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2134–2146, Online. 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*, 36(4):1234–1240.

Chenliang Li, Aixin Sun, and Anwitaman Datta. 2012. Twevent: Segment-based event detection from tweets. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 155–164, New York, NY, USA. Association for Computing Machinery.

Jiao Li, Yueping Sun, Robin J. Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J. Mattingly, Thomas C. Wiegers, and Zhiyong Lu. 2016. Biocreative V CDR task corpus: a resource for chemical disease relation extraction. *Database J. Biol. Databases Curation*, 2016.

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

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.

Hao Liu, Lirong He, Haoli Bai, Bo Dai, Kun Bai, and Zenglin Xu. 2018. Structured inference for recurrent hidden semi-markov model. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 2447–2453. AAAI Press.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26. Publisher: John Benjamins Publishing Company.

An Thanh Nguyen, Byron Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and predicting sequence labels from crowd annotations. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 299–309, Vancouver, Canada. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

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.

Alexander Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3574–3582, Red Hook, NY, USA. Curran Associates Inc.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising multi-source weak supervision for neural text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3739–3754, Online. Association for Computational Linguistics.

Esteban Safranchik, Shiying Luo, and Stephen H. Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 5570–5578. AAAI Press.

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.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? *Chinese Computational Linguistics*, page 194–206.

Christian Thiel. 2008. Classification on soft labels is robust against label noise. In *Proceedings of the 12th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part I*, KES '08, pages 65–73, Berlin, Heidelberg. Springer-Verlag.

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.

Ke M. Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. 2016. Unsupervised neural hidden Markov models. In *Proceedings of the Workshop on Structured Prediction for NLP*, pages 63–71, Austin, TX. Association for Computational Linguistics.

A. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theor.*, 13(2):260–269.

T. Wessels and Christian W. Omlin. 2000. Refining hidden markov models with recurrent neural networks. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, July 24-27, 2000, Volume 2*, pages 271–278. IEEE Computer Society.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning neural templates for text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3174–3187, Brussels, Belgium. Association for Computational Linguistics.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised NER with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2159–2169, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2020. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach.

A Technical Details

A.1 CHMM Training

Following the discussion in § 4.2, we use the *forward-backward* algorithm to calculate the smoothed marginal $\gamma_i^{(t)} \triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:T)}), i \in \{1, 2, \dots, |\mathcal{L}|\}, t \in \{1, 2, \dots, T\}$ and the expected number of transitions $\xi_{i,j}^{(t)} \triangleq p(z^{(t-1)} = i, z^{(t)} = j | \boldsymbol{x}^{(1:T)}), i, j \in \{1, 2, \dots, |\mathcal{L}|\}$.[8] $|\mathcal{L}|$ is the number of BIO formatted entity labels, which are regarded as hidden states; T is the total number of hidden steps in a sequence, which equals the number of tokens.

Defining $\alpha_i^{(t)} \triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:t)})$ and $\beta_i^{(t)} \triangleq p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = i)$, $\gamma_i^{(t)}$ and $\xi_{i,j}^{(t)}$ can be represented by α and β using the Bayes' rule and Markov assumption:

$$\begin{aligned}
\gamma_i^{(t)} &\triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:T)}) \\
&= \frac{p(\boldsymbol{x}^{(t+1:T)}, z^{(t)} = i | \boldsymbol{x}^{(1:t)})}{p(\boldsymbol{x}^{(t+1:T)} | \boldsymbol{x}^{(1:T)})} \\
&\propto p(z^{(t)} = i | \boldsymbol{x}^{(1:t)}) p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = i) \\
&= \alpha_i^{(t)} \beta_i^{(t)},
\end{aligned} \tag{12}$$

$$\begin{aligned}
\xi_{i,j}^{(t)} &\triangleq p(z^{(t-1)} = i, z^{(t)} = j | \boldsymbol{x}^{(1:T)}) \\
&\propto p(z^{(t-1)} = i | \boldsymbol{x}^{(1:t-1)}) \\
&\quad p(z^{(t)} = j | z^{(t-1)} = i, \boldsymbol{x}^{(t:T)}) \\
&\propto p(z^{(t-1)} = i | \boldsymbol{x}^{(1:t-1)}) p(\boldsymbol{x}^{(t)} | z^{(t)} = j) \\
&\quad p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = j) p(z^{(t)} = j | z^{(t-1)} = i) \\
&= \alpha_i^{(t-1)} \varphi_j^{(t)} \beta_j^{(t)} \Psi_{i,j}^{(t)}.
\end{aligned} \tag{13}$$

$\varphi_i^{(t)} \in \mathbb{R}^{|\mathcal{L}|} \triangleq p(\boldsymbol{x}^{(t)} | z^{(t)} = i)$ is the likelihood of the observation when the hidden state is i (§ 4.2).

Written in the matrix form, (12) and (13) become:

$$\boldsymbol{\gamma}^{(t)} \propto \boldsymbol{\alpha}^{(t)} \odot \boldsymbol{\beta}^{(t)}, \tag{14}$$

$$\boldsymbol{\xi}^{(t)} \propto \boldsymbol{\Psi}^{(t)} \odot (\boldsymbol{\alpha}^{(t-1)} (\boldsymbol{\varphi}^{(t)} \odot \boldsymbol{\beta}^{(t)})^{\mathsf{T}}), \tag{15}$$

where $\odot$ is the element-wise product. Note that the elements in both $\boldsymbol{\gamma}^{(t)}$ and $\boldsymbol{\xi}^{(t)}$ should sum up to 1.

The Forward Pass The filtered marginal $\alpha_i^{(t)}$ can be computed iteratively:

$$\begin{aligned}
\alpha_i^{(t)} &\triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:t)}) \\
&= p(z^{(t)} = i | \boldsymbol{x}^{(t)}, \boldsymbol{x}^{(1:t-1)}) \\
&\propto p(\boldsymbol{x}^{(t)} | z^{(t)} = i) p(z^{(t)} = i | \boldsymbol{x}^{(1:t-1)}) \\
&= \sum_j \varphi_i^{(t)} \Psi_{j,i}^{(t)} \alpha_j^{(t-1)}.
\end{aligned} \tag{16}$$

Written in the matrix form, (16) becomes

$$\boldsymbol{\alpha}^{(t)} \propto \boldsymbol{\varphi}^{(t)} \odot ({\boldsymbol{\Psi}^{(t)}}^{\mathsf{T}} \boldsymbol{\alpha}^{(t-1)}). \tag{17}$$

We initialize $\boldsymbol{\alpha}$ with $\boldsymbol{\alpha}^{(0)} = \boldsymbol{\pi}$ (§ 4.2) since we have no observation at time step 0. As $\boldsymbol{\alpha}^{(t)}$ is a probability distribution, the elements in it sum up to 1. The calculation of $\boldsymbol{\alpha}$ is the *forward pass*.

The Backward Pass In the same way, we do the *backward pass* to compute the conditional future evidence $\beta_i^{(t)} \triangleq p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = i)$:

$$\begin{aligned}
\beta_i^{(t-1)} &\triangleq p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = j) \\
&= \sum_j p(z^{(t)} = j, \boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t+1:T)} | z^{(t-1)} = i) \\
&= \sum_j [p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = j) \\
&\qquad p(\boldsymbol{x}^{(t)}, z^{(t)} = j | z^{(t-1)} = i)] \\
&= \sum_j \beta_j^{(t)} \varphi_j^{(t)} \Psi_{i,j}^{(t)}.
\end{aligned} \tag{18}$$

In the matrix form, (18) becomes:

$$\boldsymbol{\beta}^{(t-1)} = \boldsymbol{\Psi}^{(t)} (\boldsymbol{\varphi}^{(t)} \odot \boldsymbol{\beta}^{(t)}), \tag{19}$$

whose base case is

$$\beta_i^{(T)} = p(\boldsymbol{x}^{(T+1:T)} | z^{(T)} = i) = 1, \qquad \forall i \in \{1, \dots, |\mathcal{L}|\}.$$

A.2 The Maximization step for Unsupervised HMM

For traditional unsupervised HMM, the expected complete data log likelihood is maximized by updating the matrices with the approximated pseudo-statistics. different from CHMM, HMM has constant transition and emission for all time steps, *i.e.*:

$$\boldsymbol{\Psi}^{(1)} = \boldsymbol{\Psi}^{(t)}; \boldsymbol{\Phi}^{(1)} = \boldsymbol{\Phi}^{(t)}; \quad \forall t \in \{2, \dots, T\}.$$

[8]Same as § 4.2, we omit the dependency term $\boldsymbol{e}^{(1:T)}$.

For simplicity, we remove the term t for the transition and emission matrices. Suppose we are updating HMM based on one instance with t starting from 1:

$$\pi_i = \gamma_i^{(1)}; \tag{20}$$

$$\Psi_{i,j} = \frac{\sum_{t=2}^{T} \xi_{i,j}^{(t)}}{\sum_{t=2}^{T} \sum_{\ell=1}^{|\mathcal{L}|} \xi_{i,\ell}^{(t)}}; \tag{21}$$

$$\Phi_{i,j,k} = \frac{\sum_{t=1}^{T} \gamma_i^{(t)} x_{j,k}^{(t)}}{\sum_{t=1}^{T} \gamma_i^{t}}. \tag{22}$$

Note that the observation has property $0 \leq x_{j,k}^{(t)} \leq 1$ and $\sum_{j=1}^{|\mathcal{L}|} x_{j,k}^{(t)} = 1$, where $k \in \{1, \dots K\}$ is the index of the weak labeling source.

B Labeling Source Performance

The weak labeling sources of the CoNLL 2003 dataset come from Lison et al. (2020), whereas Safranchik et al. (2020) provide the sources for the LaptopReview, NCBI-Disease and BC5CDR dataset. For Safranchik et al. (2020)'s labeling sources, we apply a majority voting using their tagging results to the spans detected by their *linking rules* to convert the linking results to token annotations. In consideration of the training time and resource consumption, we only adopt a subset of the labeling sources provided by the authors. The performance of the labeling sources is presented in the tables below.

source name	precision	recall	f1
CoreDictionaryUncased	81.03	41.41	5.48
CoreDictionaryExact	80.69	17.18	28.32
CancerLike	34.88	1.58	3.02
BodyTerms	68.52	3.90	7.38
ExtractedPhrase	97.12	32.03	48.18

Table 4: The performance of the labeling sources used in the NCBI-Disease dataset.

source name	precision	recall	f1
DictCore-Chemical	91.81	29.55	44.7
DictCore-Chemical-Exact	85.88	3.16	6.1
DictCore-Disease	81.57	26.32	39.8
DictCore-Disease-Exact	81.4	1.09	2.16
Organic Chemical	92.67	30.07	45.4
Disease or Syndrome	77.36	11.67	20.28
PostHyphen	84.47	08.07	14.74
ExtractedPhrase	86.8	17.96	29.76

Table 5: The performance of the labeling sources used in the BC5CDR dataset.

source name	precision	recall	f1
CoreDictionary	72.63	51.61	60.34
iStuff	26.67	0.61	1.2
ExtractedPhrase	97.45	29.25	45.0
ConsecutiveCapitals	35.29	0.92	1.8

Table 6: The performance of the labeling sources used in the LaptopReview dataset.

source name	precision	recall	f1
BTC+c	61.56	46.35	52.88
SEC+c	39.54	24.59	30.32
core_web_md+c	69.53	60.04	64.44
crunchbase_cased	38.26	5.59	9.76
crunchbase_uncased	37.88	6.2	10.66
doc_majority_cased	65.81	40.21	49.92
doc_majority_uncased	61.69	40.17	48.66
full_name_detector	87.79	11.33	20.06
geo_cased	68.16	15.35	25.06
geo_uncased	65.1	18.89	29.28
misc_detector	85.14	21.51	34.34
wiki_cased	75.27	32.65	45.54
wiki_uncased	72.26	35.61	47.7

Table 7: The performance of the labeling sources used in the CoNLL 2003 dataset.

Please refer to Lison et al. (2020) for the information about the construction of the labeling sources on the CoNLL 2003 dataset; please refer to Safranchik et al. (2020) for the labeling sources on other three datasets.

C Hyper-Parameters

The experiments are conducted on one GeForce RTX 2080 Ti GPU. For NCBI-Disease, BC5CDR and LaptopReview datasets, CHMM is pre-trained for 5 epochs and trained for 20 epochs. The learning rates for these three datasets are 5×10^{-4}, 10^{-3} and 10^{-4}, respectively, and the batch sizes are 64, 64 and 128. In phase I, BERT-NER is trained with the default learning rate (5×10^{-5}) for 100 epochs. The batch sizes are 8, 8, and 48, respectively. Note that for LaptopReview, the maximum length limitation of BERT-NER is set to 128 whereas the limitation is 512 for the other two datasets. In phase II, we use half the learning rate with 20 epochs for each loop.

For CoNLL 2003, CHMM has the same number of training epochs as for other datasets. The batch size is 32, and the learning rate is 10^{-5}. BERT-NER has a maximum sequence length of 256. It is trained for 15 epochs in phase I and 5 epochs in phase II. Other hyper-parameters are identical to other BERT-NER models'.