Sparse Conditional Hidden Markov Model for Weakly Supervised Named Entity Recognition

Yinghao Li
Georgia Institute of Technology
Atlanta, Georgia, USA
yinghaoli@gatech.edu

Le Song
BioMap, MBZUAI
Beijing, China
le.song@mbzuai.ac.ae

Chao Zhang
Georgia Institute of Technology
Atlanta, Georgia, USA
chaozhang@gatech.edu

ABSTRACT

Weakly supervised named entity recognition methods train *label models* to aggregate the token annotations of multiple noisy labeling functions (LFs) without seeing any manually annotated labels. To work well, the label model needs to contextually identify and emphasize well-performed LFs while down-weighting the under-performers. However, evaluating the LFs is challenging due to the lack of ground truths. To address this issue, we propose the *sparse conditional hidden Markov model* (Sparse-CHMM). Instead of predicting the entire emission matrix as other HMM-based methods, Sparse-CHMM focuses on estimating its diagonal elements, which are considered as the reliability scores of the LFs. The sparse scores are then expanded to the full-fledged emission matrix with pre-defined expansion functions. We also augment the emission with weighted XOR scores, which track the probabilities of an LF observing incorrect entities. Sparse-CHMM is optimized through unsupervised learning with a three-stage training pipeline that reduces the training difficulty and prevents the model from falling into local optima. Compared with the baselines in the Wrench benchmark, Sparse-CHMM achieves a 3.01 average F_1 score improvement on 5 comprehensive datasets. Experiments show that each component of Sparse-CHMM is effective, and the estimated LF reliabilities strongly correlate with true LF F_1 scores.

CCS CONCEPTS

• **Computing methodologies → Information extraction**.

KEYWORDS

Hidden Markov Model, Weak Supervision, Information Extraction, Named Entity Recognition

ACM Reference Format:
Yinghao Li, Le Song, and Chao Zhang. 2022. Sparse Conditional Hidden Markov Model for Weakly Supervised Named Entity Recognition. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539247

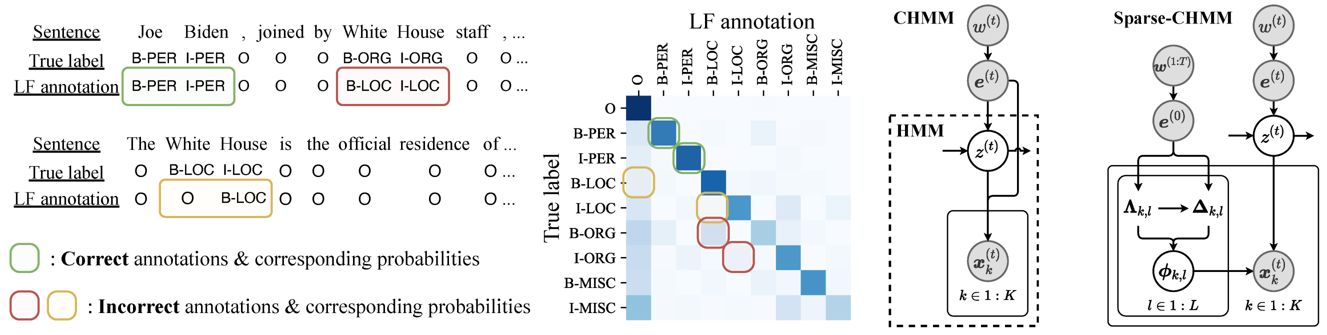
1 INTRODUCTION

Named entity recognition (NER), a task that identifies pre-defined named entities, is fundamental for extracting information from unstructured text data. For example, to facilitate material synthesis, material scientists often need to extract entities such as materials, properties, and catalysts from research articles [9]. NER also powers tasks such as question answering and knowledge graph construction. Currently, prevailing NER methods are fully supervised models trained on a substantial amount of human-annotated data. However, annotating sentences with named entity labels is difficult, expensive, and time-consuming [2, 31]. The manually annotated dataset is also hard to expand—to insert new entity types or append new sentences, annotators must go over each sample, the effort of which is proportional to the data size. *Weak supervision* is proposed in recent works [7, 21] to address this difficulty. Instead of human annotations, weakly supervised methods generate multiple sets of *weak annotations* using labeling functions (LFs, *a.k.a.* weak supervision sources). They can be knowledge bases, domain-specific dictionaries, pattern matching regular expressions or out-of-domain supervised models [16]. Constructing LFs is significantly cheaper than annotating tokens manually. And the weak annotations are expandable—the LFs can automatically annotate incoming sentences without any extra human labor.

Nonetheless, it is challenging to design *label models* which aggregate the LF annotations without seeing any ground-truth labels. LFs are usually simple, and their annotations often suffer from incompleteness (low coverage/recall) and inaccuracy (low precision). Moreover, the LFs can conflict with each other, generating contradictory results [23]. Without true labels, solving the conflicts and reducing the annotation noise become a tricky task.

Some weakly supervised NER methods such as majority voting (MV) and Snorkel [21] classify each token in the sentence independently. Such approaches neglect the complex inter-word relationship of natural language. Recent works [14, 16, 18, 19, 25] parameterize the label sequences with graphical models, regarding the unobserved "true labels" as latent variables and inferring them by optimizing the probabilities of the observed weak annotations through unsupervised learning. Nguyen et al. [18], Safranchik et al. [25] and Lison et al. [16] apply the hidden Markov models (HMMs) to capture the token dependency with the transition matrix. However, the Markov property constrains HMMs' modeling of such dependency to only consecutive tokens meanwhile largely ignoring sentence semantics. There are attempts to integrate neural networks (NNs) with graphical models [14, 19]. Li et al. [14] propose the conditional HMM (CHMM) that uses BERT token embeddings [5] to predict HMMs' transition and emission probabilities with NNs. As each token embedding encodes the contextual semantics of the whole sentence, CHMM incorporates more comprehensive context information and alleviates the Markov constraint. A similar approach is DWS [19], which substitutes HMM by the conditional

KDD '22, August 14–18, 2022, Washington, DC, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539247


Sentence Joe Biden , joined by White House staff , ...
True label B-PER I-PER O O O B-ORG I-ORG O O ...
LF annotation B-PER I-PER O O O B-LOC I-LOC O O ...
Sentence The White House is the official residence of ...
True label O B-LOC I-LOC O O O O O ...
LF annotation O O B-LOC O O O O O ...
: Correct annotations & corresponding probabilities
: Incorrect annotations & corresponding probabilities
LF annotation
True label
O B-PER I-PER B-LOC I-LOC B-ORG I-ORG B-MISC I-MISC
CHMM
HMM
Sparse-CHMM
$w^{(t)}$
$e^{(t)}$
$z^{(t)}$
$x_k^{(t)}$
$k \in 1:K$
$w^{(1:T)}$
$e^{(0)}$
$\Lambda_{k,l}$
$\Delta_{k,l}$
$\phi_{k,l}$
$l \in 1:L$

(a) The annotations and the corresponding true emission matrix of *one* example LF

(b) Model structures of CHMM and Sparse-CHMM

Figure 1: (a) shows an example of the weak annotations for weakly supervised NER and HMM's corresponding true emission pattern; (b) illustrates Sparse-CHMM's model structure. Λ_k and Δ_k in (b) respectively parameterize the diagonal and off-diagonal emission elements in (a), which are LF's probabilities of generating correct and incorrect labels.

random field (CRF) and predicts the latent variables instead of their transitions. The problem with both CHMM and DWS is that they directly predict all elements in the emission matrices from the embeddings. When the number of LFs or labels is large, doing so requires a myriad of network parameters. As the optimization is non-convex, this creates many local optima, making the model hard to train. Besides, the training and inference of a large NN are much more computationally demanding, which hurts model efficiency.

In this work, we propose Sparse-CHMM. Maintaining the transition scheme, Sparse-CHMM enhances CHMM by restricting its emission process with the empirical structure of an emission matrix (Figure 1a). Generally, the diagonal of the emission matrix is its most essential component, which represents the reliability of the LF annotations. Considering this fact, Sparse-CHMM first focuses on estimating the LF reliability scores instead of the entire emission matrix. Then it expands the sparse scores to the full-fledged emission matrix by placing the scores at the diagonals and filling up other vacancies with our designed expansion functions (section 3.2.1). Compared with CHMM, Sparse-CHMM features sparser emission prediction process with much fewer parameters to learn, making the model easier to optimize and more efficient to train.

To enhance Sparse-CHMM's estimation of off-diagonal emission elements, we introduce another component—the weighted XOR (WXOR) scores (section 3.2.2). The off-diagonal elements model LFs' probabilities of generating incorrect entity annotations. These probabilities are typically small but sometimes comparable with or larger than the diagonal elements. In such cases, using the diagonal-oriented reliability expansion functions alone is biased and inflexible. To address this issue, we leverage LF annotations and the predicted LF reliabilities to calculate LFs' mutual disagreement frequencies as WXOR scores to refine off-diagonal emission elements.

Considering the amount of uncertainty in the emission, we treat the function-induced deterministic values as priors, with which a Dirichlet distribution is parameterized. We then sample our final emission probabilities from the distribution (section 3.2.3). Previous works [3, 10] show that incorporating the probability model expands the search space, regularizes the gradient flow, and further prevents the model from being trapped by local optima.

In addition, to improve learning Sparse-CHMM's inter-correlated components, we introduce a three-stage training strategy. Each stage optimizes only a subset of model parameters (section 3.3.3) to reduce mutual interference and promote the training stability.

In summary, our contributions include:

- proposing Sparse-CHMM to estimate LF reliability scores from sentence embeddings, and expand the scores to sparsely predicted emission matrices with our designed expansion functions;
- modeling the probabilities of an LF generating incorrect annotations with the WXOR scores to augment the estimation of emission off-diagonal elements;
- a three-stage training strategy and the Dirichlet sampling process to improve training stability and model robustness; and
- thorough experiments on 5 comprehensive datasets with SOTA baselines that demonstrate the superiority of Sparse-CHMM, and ablation studies that justify the design of model components.

The code and data are available at github.com/Yinghao-Li/Sparse-CHMM for reproducibility.

2 PROBLEM DEFINITION

In this section, we define the weakly supervised NER task. Suppose we have T tokens $\boldsymbol{w}^{(1:T)}$ in the input sentence and a set of candidate entities $\mathcal{E}$ with size E, NER models assign one entity label to each token $\boldsymbol{w}^{(t)}, t \in 1:T$. Using the BIO tagging scheme, the label set is $\mathcal{L} = \{\texttt{O}\} \cup \{\texttt{B-ent}, \texttt{I-ent}\}_{\text{ent}\in\mathcal{E}}$ with the size of $L = 2E+1$. Label "O" indicates that the current token belongs to no pre-defined entities, which is always indexed by 1 in the following discussion. An example of NER labels is shown in Figure 1a.

For weakly supervised NER, we have K independent LFs, each providing a sequence of weak annotations $\boldsymbol{x}_k^{(1:T)}$. $\boldsymbol{x}_k^{(t)} \in \{0,1\}^L$ is one-hot over the label set $\mathcal{L}$. Label models aim to approximate the ground-truth labels $\boldsymbol{y}^{(1:T)} \in \mathcal{L}^T$ with latent states $\boldsymbol{z}^{(1:T)} \in \mathcal{L}^T$ given the input tokens and weak annotations $\{\boldsymbol{w}^{(1:T)}, \boldsymbol{x}_{1:K}^{(1:T)}\}$.

In the following discussion, we do not distinguish the label string and label index, and uniformly represent them by integers l, i, or $j \in 1:L$. In addition, we focus on one sentence and omit the sentence index $m \in 1:M$ unless specified otherwise.

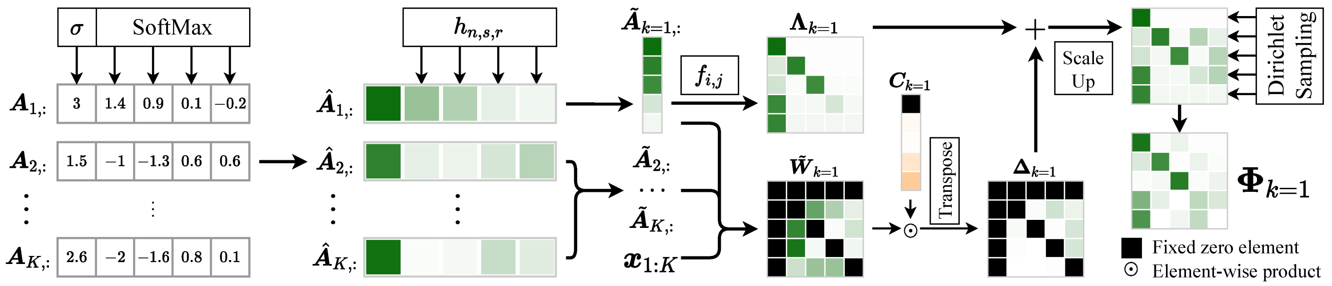


Figure 2: Emission construction pipeline for LF $k = 1$ with $L = 5$ labels. Darker colors indicate larger values in range $[0, 1]$.

3 METHOD

Sparse-CHMM has three main components: 1) the transition matrix Ψ (section 3.1), 2) the *emission base prior* matrix Λ (section 3.2.1), and 3) the *emission addon prior* matrix Δ (section 3.2.2). The transition matrix controls the probability of moving from one latent state to another. Λ and Δ are the components of the emission matrix Φ (section 3.2.3), each element in which $\Phi_{k,i,j} \triangleq p(x_{k,j}^{(t)} = 1|z^{(t)} = i)$ defines the probability of LF k observing label j when the true label is i. Λ focuses on the emission diagonals, whereas Δ refines the off-diagonal values. In section 3.3, we describe the training strategy and present a three-stage pipeline that increases training stability and maximizes the model performance.

3.1 Transition

Same as Li et al. [14], we directly predict the *token-wise* transition probabilities $\Psi_{i,j}^{(t)} \triangleq p(z^{(t)} = j|z^{(t-1)} = i, \boldsymbol{e}^{(t)})$ from the BERT *token* embeddings $\boldsymbol{e}^{(t)}$. Specifically, we input the embeddings into one fully connected (FC) layer of NN and reshape the output vectors into $L\times L$ matrices. A SoftMax function is applied along the columns to make each row a point in an L-dimensional probability simplex.

$$S^{(t)} \in \mathbb{R}^{L\times L} = \text{reshape}(\text{FC}(\boldsymbol{e}^{(t)}));$$
$$\Psi_{l,:}^{(t)} \in (0,1)^L = \text{SoftMax}(S_{l,:}^{(t)}).$$

Here $S^{(t)}$ is an intermediate variable, and l is the row label index.

3.2 Emission

Instead of predicting every element in the emission matrices $\Phi^{(t)} \in [0,1]^{K\times L\times L}$, Sparse-CHMM predicts LF reliabilities $\tilde{A} \in [0,1]^{K\times L}$ with NN and expands it to the base prior $\Lambda \in [0,1]^{K\times L\times L}$ through pre-defined functions (section 3.2.1). From $\tilde{A}$ and LF observations $\boldsymbol{x}$, we also calculate the WXOR scores $\tilde{W} \in [0,1]^{K\times L\times L}$ that track the probabilities of LFs' giving incorrect annotations. $\tilde{W}$ is then scaled by another NN predicted matrix $C \in [0,1]^{K\times L}$ to form the addon prior Δ (section 3.2.2). We sample the emission matrix Φ from a Dirichlet distribution parameterized by Λ and Δ to facilitate model training (section 3.2.3). The pipeline is illustrated in Figure 2.

Compared with the transition, the emission variance is much smaller *w.r.t.* the input tokens. So we predict the *sentence-level* emission-related variables from the BERT *sentence* embedding $\boldsymbol{e}^{(0)}$ [5], which deprecates their token indicator "$\cdot^{(t)}$".

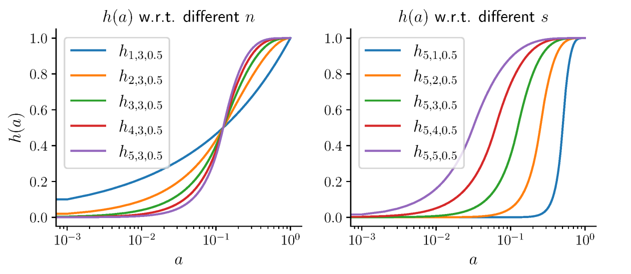


Figure 3: Reliability scaling function $h_{n,s,r}(a)$ *w.r.t.* its exponential terms n and s. $s > 1$ scales up all values, whereas $n > 1$ suppresses small values but augments large ones.

3.2.1 *Labeling Function Reliabilities and Emission Base Prior.*

For LF k, the diagonal elements of the emission matrix

$$\Phi_{k,l,l} \triangleq p(x_{k,l}^{(t)} = 1|z^{(t)} = l, \boldsymbol{e}^{(0)}), \quad l \in 1:L,$$

as shown in Figure 1a, has a distinctive physical meaning: the probabilities of LF k observing the correct labels, *i.e.*, the reliabilities of LF k. In other words, we can estimate the performance of LF k given its emission matrix Φ_k, or vice versa: *we can construct a plausible emission matrix given the knowledge of how an LF performs.*

With this concept established, we start by predicting the reliability logits $A \in \mathbb{R}^{K\times L}$ from the sentence embedding:

$$A = \text{reshape}(\text{FC}(\boldsymbol{e}^{(0)})),$$

which is to be placed at the diagonal of Φ after normalization.[1] However, as LFs observe more label O (indexed by 1) than other labels, the model tend to emphasize the corresponding weight $\Phi_{k,l,1} \triangleq p(x_{k,1}^{(t)} = 1|z^{(t)} = l, \boldsymbol{e}^{(0)})$ in the emission. Because the vector $\Phi_{k,l}$ is from a simplex and sums up to 1, large $\Phi_{k,l,1}$ results in unreasonably small diagonal values $\Phi_{k,l,l}$. Fortunately, this can be fixed by applying SoftMax along the LFs:

$$\hat{A}_{:,l} = \text{SoftMax}(A_{:,l}), \quad l \geq 2. \tag{1}$$

It guarantees that at least one LF has a high score, preventing the model from neglecting all weak annotations simultaneously. Since $A_{k,1}$ is the emission confidence to O itself, we only need to constrain its value to $(0,1)$ through the element-wise sigmoid function σ:

$$\hat{A}_{:,1} = \sigma(A_{:,1}).$$

[1]On datasets with more than one entity types, we adopt entity-level reliability scores, *i.e.*, $A_{k,\texttt{B-ent}} = A_{k,\texttt{I-ent}}$. This reduces the model size and training difficulty.

However, SoftMax in (1) enforces $\sum_k \hat{\boldsymbol{A}}_{k,l} = 1$. This correlation establishes a *fixed-size* pool from which reliability scores are drawn, which harms the model flexibility as it fails the cases where multiple LFs are confident, or no LF is confident. To de-correlate the scores, we introduce a piecewise scaling function (illustrated in Figure 3):

$$\tilde{A}_{k,l} = h_{n,s,r}(\hat{A}_{k,l});$$
$$h_{n,s,r}(a) = \begin{cases} \frac{1}{r^{n-1}} a & a^{\frac{1}{s}} < r; \\ -\frac{1}{(1-r)^{(n-1)}}(1-a^{\frac{1}{s}})^n + 1 & a^{\frac{1}{s}} \geq r. \end{cases} \tag{2}$$

It calibrates the SoftMax output with scales defined by exponents n and s. To enable more subtle control of the scores, we utilize the split point $r \in [0,1]$ to segment the input domain into upper and lower halves, each scaled differently.[2]

We then expand $\tilde{\boldsymbol{A}}$ to the *emission base prior* matrix $\boldsymbol{\Lambda} \in [0,1]^{K \times L \times L}$ through expansion functions defined according to the latent state $z^{(t)} = i$ and the observation $x^{(t)}_{k,j} = 1$:

$$\Lambda_{k,i,j} = f_{i,j}(\tilde{A}_{k,i}); \quad \forall i,j \in 1:L,$$
$$f_{i,j}(a) = \begin{cases} a & i = j; \\ \frac{1}{L-1}(1-a) & i = 1, j \geq 2; \\ g_{n,r}(a) & i \geq 2, j = 1; \\ \frac{1}{L-2}(1-a-g_{n,r}(a)) & i \geq 2, j \geq 2, i \neq j, \end{cases} \tag{3}$$
$$g_{n,r}(a) = \begin{cases} \frac{2-L}{(n-1)r^n - nr^{n-1}} a^n + (1-L)x + 1 & a \leq r; \\ \frac{g_{n,r}(r)}{r-1} a - \frac{g_{n,r}(r)}{r-1} & a > r, \end{cases} \tag{4}$$

where n is the exponential term and r is the split point. The reliability scores $\tilde{\boldsymbol{A}}_k$ of LF k are placed at the diagonal of $\boldsymbol{\Lambda}_k$ (*i.e.*, function $f_{i=j}$). When the latent state is 0 ($z^{(t)} = i = 1$), the probabilities of emitting to non-0 $p(\boldsymbol{x}^{(t)}_{k,j} = 1 | z^{(t)} = 1, \boldsymbol{e}^{(0)})$, $\forall j \geq 2$ are uniform and sum up to $1 - \tilde{A}_{k,1}$. When the latent state is not 0 ($z^{(t)} = i \geq 2$), an LF statistically inclines toward observing 0 than other entities, making emit-to-0 probabilities $p(\boldsymbol{x}^{(t)}_{k,1} = 1 | z^{(t)} = i, \boldsymbol{e}^{(0)})$, $\forall i \geq 2$ larger and more important than other off-diagonal values $p(\boldsymbol{x}^{(t)}_{k,j} = 1 | z^{(t)} = i, \boldsymbol{e}^{(0)})$, $\forall i,j \geq 2, i \neq j$, as shown in Figure 1a. Therefore, we specify $g_{n,r}$ to emphasize emit-to-0 probabilities.[2] With small reliability scores, an LF is less confident about choosing which label, so the Uniform off-diagonal non-0 values are set close to the diagonals. When the scores are above threshold r, the LF is considered confident enough to abate the emission probabilities to other entities. The exponent n controls how fast the emit-to-0 probabilities decrease, *i.e.*, how close off-diagonal values to the diagonal before r. The functions f and g are illustrated in Figure 4.

3.2.2 *Weighted XOR and Emission Addon Prior.*

In contrast to the diagonals $\Phi_{k,l,l}$, the off-diagonal values $\Phi_{k,i,j}; i,j \geq 2; i \neq j$ are the probabilities of LF k *misclassifying* label i as j. Such probabilities are generally insignificant. But in some cases, they can be prominent (as the example shown in Figure 1a) and affects the model performance. The base prior $\boldsymbol{\Lambda}$ assigns equal weights to all off-diagonals and fails to adjust to such cases.

Therefore, we develop the WXOR scores $\tilde{\boldsymbol{W}} \in [0,1]^{K \times L \times L}$ to capture the misclassification probabilities. Given the annotations of LF k and other LFs $k' \in 1:K \backslash k$, a WXOR logit is specified between the query label $l^{\text{query}} \in 2:L$ and the target label $l^{\text{tgt}} \in 2:L$:

$$W^{(t)}_{k,l^{\text{query}},l^{\text{tgt}}} = (1 - \tilde{A}_{k,l^{\text{query}}}) x^{(t)}_{k,l^{\text{query}}} \sum_{k'=1}^{K} \tilde{A}_{k',l^{\text{tgt}}} x^{(t)}_{k',l^{\text{tgt}}};$$
$$\boldsymbol{W}^{(t)}_{k,1,:} = \boldsymbol{W}^{(t)}_{k,:,1} = 0; \quad W^{(t)}_{k,l,l} = 0, \ \forall l \in 1:L. \tag{5}$$

It depicts the likelihood that LF k makes mistakes in its observation of l^{query} *w.r.t.* the other LFs at time-step t. $1 - \tilde{A}_{k,l^{\text{query}}}$ is the *uncertainty level* of LF k to its observation of l^{query}; $\tilde{A}_{k',l^{\text{tgt}}}$ is the confidence of LF k' to l^{tgt}; and the binary $x^{(t)}_{k,l}$ decides whether l is observed. $W^{(t)}_{k,l^{\text{query}},l^{\text{tgt}}}$ is large when LF k observes l^{query} but is uncertain, while other LFs are confident about observing l^{tgt}. As $W^{(t)}_{k,l^{\text{query}},l^{\text{tgt}}}$ is non-zero *i.f.f.* LF k does but other LFs do not observe l^{query}, it is similar to the XOR gate and hence gets its name. Label 0 and the diagonal need no WXOR as these elements are well-defined in $\boldsymbol{\Lambda}$, so the corresponding values in $\boldsymbol{W}$ are fixed to 0.

$\boldsymbol{W}^{(t)}$ tensors are then summed up and normalized to form the ubiquitous WXOR: for M sentences with length $T_m, m \in 1:M$ each,

$$\hat{W}_{k,l^{\text{query}},l^{\text{tgt}}} = \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} W^{(t)}_{m,k,l^{\text{query}},l^{\text{tgt}}}}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} x^{(t)}_{m,k,l^{\text{query}}}}.$$

Combining $\hat{\boldsymbol{W}}$ with the emission base prior $\boldsymbol{\Lambda}$ is also nontrivial. Straightforward summation is unpractical since the relative scale of $\hat{\boldsymbol{W}}$ and $\boldsymbol{\Lambda}$ is obscure due to the summation in (5). In regard of this, we leverage another scaling factor $\boldsymbol{C} \in (0,1)^{K \times L}$ predicted from the BERT sentence embedding through one layer of NN:

$$\boldsymbol{C} = \text{reshape}(\sigma(\text{FC}(\boldsymbol{e}^{(0)}))).$$

Its domain $(0,1)$, however, enables $\boldsymbol{C}$ to down-scale values but not up-scale them. Consequently, we need to properly adjust the elements of $\hat{\boldsymbol{W}}$ so that they have large enough pre-scaling values:

$$\tilde{\boldsymbol{W}}_{k,:,l^{\text{tgt}}} = \text{SoftMax}(\hat{\boldsymbol{W}}_{k,:,l^{\text{tgt}}}).$$

With the established matrices, we construct the *emission addon prior* $\boldsymbol{\Delta} \in [0,1]^{K \times L \times L}$ by re-scaling the rows of $\tilde{\boldsymbol{W}}_k$:

$$\Delta_{k,:,l} = C_{k,l} \times \tilde{\boldsymbol{W}}_{k,l,:}. \tag{6}$$

Notice that in $\boldsymbol{\Delta}_k$, the target labels are at the first axis and the query labels second, different from $\tilde{\boldsymbol{W}}_k$. As the target labels are

[2]Please refer to appendix D for design principles.

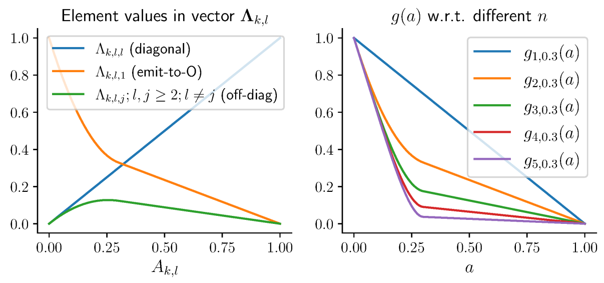


Figure 4: Left: the elements in $\Lambda_{k,l}, \forall k,l$ *w.r.t.* the reliability score $A_{k,l}$ with $L = 5, r = 0.3, n = 2$. Right: $g_{n,r}(a)$ *w.r.t.* exponent n. Larger n decreases emit-to-0 probabilities faster.

essentially the latent states we want to predict, this unifies the physical meaning of $\tilde{\boldsymbol{W}}$ and the emission matrix $\boldsymbol{\Phi}$.

3.2.3 Emission Matrix Sampling. Instead of using the deterministic priors $\boldsymbol{\Lambda}$ and $\boldsymbol{\Delta}$ directly, we choose to sample the latent emission variables from the Dirichlet distribution parameterized by the priors. This is a common practice in graphical models considering that proper latent distributions are more representative and robust to noise than the deterministic priors [3, 10]. The random sampling helps the model escape the saddle points or local optima. In addition, Dirichlet distribution samples from probability simplex without requiring its concentration parameters to sum up to one, which makes the parameters selection more flexible.

To construct the concentration parameters $\Omega \in \mathbb{R}_+^{K\times L\times L}$, we add the base prior and the addon prior together and scale the results:

$$\Omega = \nu^{\text{expan}} \times (\boldsymbol{\Lambda} + \boldsymbol{\Delta}) + \nu^{\text{base}}.$$

$\nu^{\text{base}} \in \mathbb{R}_+$ and $\nu^{\text{expan}} \in \mathbb{R}_+$ controls the minimum concentration value and the concentration range, respectively. Larger ν_{expan} results in smaller sampling variance. Each row of the emission matrix $\boldsymbol{\Phi}_k$ of LF k is sampled independently from the distribution:

$$\boldsymbol{\Phi}_{k,l} \sim \text{Dir}(\Omega_{k,l}).$$

We use pathwise derivative estimators developed by Jankowiak and Obermeyer [8] to push the gradient back through the Dirichlet.

Notice that we only apply Dirichlet sampling when it requires backpropagation. On other occasions, such as validation and test, the samples are substituted by the mean of the Dirichlet distribution.

3.3 Model Training

Sparse-CHMM is an unsupervised model whose optimization does not leverage any ground-truth labels $\boldsymbol{y}$ but only the weak labels $\boldsymbol{x}$. Section 3.3.1–3.3.3 describe Sparse-CHMM's training strategy, whereas section 3.3.4 analyzes the inference complexity.

3.3.1 Model Pre-training. A good initialization is critical for HMMs to gain good performance. However, as Sparse-CHMM contains several NNs, assigning proper initial values to model parameters is unrealistic. Thus, we adopt the approach used by Li et al. [14], which pre-trains the neural networks by minimizing the Euclidean distances between the predicted transitions and emissions and the initial matrices $\boldsymbol{\Psi}^*$ and $\boldsymbol{\Phi}^*$ that are from the observation statistics. The objective is the mean squared error (MSE) loss:

$$\ell_{\text{MSE}} = \frac{1}{K}\sum_{k=1}^{K} \|\boldsymbol{\Phi}_k - \boldsymbol{\Phi}^*\|_F^2 + \frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{\Psi}^{(t)} - \boldsymbol{\Psi}^*\|_F^2, \quad (7)$$

where $\|\cdot\|_F$ is the Frobenius norm. Different from [14] or [16], we require no prior knowledge of LF performance.

3.3.2 Training Objective. Same as CHMM [14], Sparse-CHMM is trained using the generalized EM algorithm. Specifically, we leverage the gradient ascent method in the M step to optimize the expected complete data log likelihood computed in the E step:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) \triangleq \mathbb{E}_{\boldsymbol{z}}[\ell_c(\boldsymbol{\theta})|\boldsymbol{x}, \boldsymbol{\theta}^{\text{old}}]. \quad (8)$$

The expectation is defined over the hidden states given LF observations and the current model parameters $p(\boldsymbol{z}^{(1:T)}|\boldsymbol{x}^{(1:T)}, \boldsymbol{\theta}^{\text{old}})$.

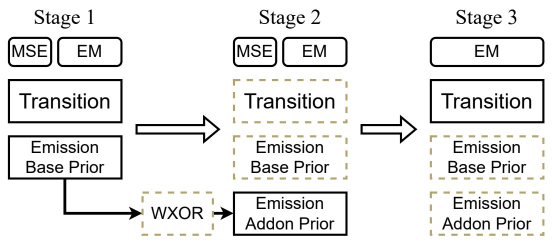


Figure 5: Sparse-CHMM training procedure. The model parameters of the dotted squares are frozen. MSE and EM are optimization approaches associated with the model pre-training and training steps.

The data log likelihood $\ell_c(\boldsymbol{\theta})$ is defined as:

$$\ell_c(\boldsymbol{\theta}) \triangleq \log p(z^{(0:T)}, \boldsymbol{x}^{(1:T)}|\boldsymbol{\theta}, \boldsymbol{e}^{(0:T)}) = \log p(z^{(0)}) + \sum_{t=1}^{T} \log p(z^{(t)}|z^{(t-1)}, \boldsymbol{e}^{(t)}) + \sum_{t=1}^{T} \log p(\boldsymbol{x}^{(t)}|z^{(t)}, \boldsymbol{e}^{(0)}), \quad (9)$$

of which the conditional independency comes from the Markov assumption. The computation of (8) largely aligns with CHMM. One difference is the dependency of the emission evidence:

$$\varphi_l^{(t)} \triangleq p(\boldsymbol{x}^{(t)}|z^{(t)} = l, \boldsymbol{e}^{(0)}) = \prod_{k=1}^{K}\sum_{j=1}^{L} \Phi_{k,l,j} x_{k,j}, \quad (10)$$

which in our case is sentence-level $\boldsymbol{e}^{(0)}$, whereas in CHMM is token-level $\boldsymbol{e}^{(t)}$. This essentially is attributed to the different approaches to obtain the emission matrix $\boldsymbol{\Phi}$ (section 3.2). The computation of (8) and other forward inference details are in appendix A.

3.3.3 Training Procedure. As the computation of the emission is complicated and inter-dependent, we adopt a three-stage training strategy to allow sufficient training of each model component, as shown in Figure 5. It decouples the optimization of the model components while keeping the training efficient.

Stage 1 is focused on optimizing the transition matrix $\boldsymbol{\Psi}$ and the emission base prior $\boldsymbol{\Lambda}$, excluding the addon prior by setting $\boldsymbol{\Delta} = 0$. The reason is straightforward: calculating the WXOR scores $\tilde{\boldsymbol{W}}$ requires high-quality reliability scores $\tilde{\boldsymbol{A}}$, which are obtained by optimizing $\boldsymbol{\Lambda}$ (section 3.3). Stage 2 trains the addon prior $\boldsymbol{\Delta}$, leaving $\boldsymbol{\Psi}$ and $\boldsymbol{\Lambda}$ frozen. This stage aims to search for the best scaling factors $\boldsymbol{C}$ for $\tilde{\boldsymbol{W}}$, and freezing the irrelevant parameters relieves the training pressure. $\tilde{\boldsymbol{W}}$ is calculated right after stage 1 with all training and validation instances and remains constant for stages 2 and 3.

Stage 3 is inspired by our empirical discovery about HMMs. We find that if we only optimize the transition matrix $\boldsymbol{\Psi}$ with the emission fixed, generally to the true values, the model performance can be improved. The true emission is the statistics of the annotations $\boldsymbol{x}_k$ of LF k given ground-truth labels $\boldsymbol{y}$:

$$\Phi_{k,i,j}^{\text{true}} = p(\boldsymbol{x}_k|\boldsymbol{y}) \triangleq \frac{\sum_{m=1}^{M}\sum_{t=1}^{T_m} \mathbb{I}(x_{m,k,j}^{(t)} = 1, y_m^{(t)} = i)}{\sum_{m=1}^{M}\sum_{t=1}^{T_m} \mathbb{I}(y_m^{(t)} = i)},$$

where $\mathbb{I}(\cdot)$ is the indicator function. Consequently, we believe that continuing training the transition of Sparse-CHMM for several

Table 1: Emission complexity of each epoch.

	CHMM	Sparse-CHMM
# NN PRM	$d^{\text{emb}} \times K \times L^2$	$2 \times d^{\text{emb}} \times K \times L$
NN CPL	$O(M \times T \times d^{\text{emb}} \times K \times L^2)$	$O(M \times d^{\text{emb}} \times K \times L)$

PRM is short for "parameters"; CPL is short for "complexity".
M is the number of training sentences, T is the average of sentence lengths, and d^{emb} is the dimension of the BERT embeddings.

more epochs with the emission frozen would lead to a similar performance improvement.

In addition, stages 1, 2, and 3 use different pre-training strategies. The pre-training of stage 1 is the same as section 3.3.1. Stage 2's pre-training is to initialize the NN parameters associated with the addon prior Δ only, so we drop the transition part of the MSE loss and substitute the target emission by $\Phi' = \lambda\Phi^* + \frac{(1-\lambda)}{M} \sum_{m=1}^{M} \Phi_m^{(I)}$ to take advantage of stage 1's results. Here $\lambda \in [0, 1]$ is the weight of observation statistics, which is fixed to 0.2 in our experiments. $\Phi_m^{(I)}$ is the optimized emission from stage 1. Stage 3 successes all model parameters from the previous stage and thus has no pre-training.

3.3.4 Complexity Analysis. Compared with CHMM, Sparse-CHMM significantly reduces the training resource consumption by predicting sparse emission elements. The emission NN parameter number and computation complexity are shown in Table 1, where the transition attributes are not presented because they are the same for both models. The factor 2 for Sparse-CHMM comes from matrices $\tilde{A}$ and C. We can see that Sparse-CHMM reduces the emission NN parameter number to $2/L$ of CHMM and the complexity to $1/(T \times L)$, which are substantial when the number of entity labels L is large. The complexity of other emission elements, *i.e.*, (2)–(5), is negligible because they do not contain the embedding dimension d^{emb}, which is much larger than other terms. Calculating the WXOR scores can be as complex as $O(M \times T \times K^2 \times L^2)$, but they are calculated only once at stage 2 and stay fixed henceforth.

4 EXPERIMENTS

4.1 Experiment Setup

4.1.1 Datasets. We consider 5 NER datasets from the generic domains to the biomedical and chemical domains: 1) **CoNLL 2003** (English subset) [27] labels 22,137 sentences from the Reuters news stories with 4 entity types, including `PER`, `LOC`, `ORG` and `MISC`. 2) The **LaptopReview** dataset [20] consists of 3,845 sentences of laptop reviews. The laptop-related `Terms` are regarded as named entities. 3) The **NCBI-Disease** dataset [6] contains 793 PubMed abstracts annotated with `Disease` mentions. 4) **BC5CDR** [13] consists of 1,500 PubMed articles with `Chemical` and `Disease` entities. 5) **OntoNotes 5.0** [29] is a very large dataset containing 143,709 sentences labeled with 18 fine-grained entity types.

We use the LFs included in the Wrench benchmark platform [31] for all datasets. Table 2 shows the dataset statistics, including the number of entities and labeling functions. The performance of the LFs on the test dataset is listed in appendix B.

4.1.2 Baselines. We compare our model with the following Wrench baselines: 1) **Majority Voting** (MV) returns the label that has been

Table 2: Dataset statistics.

	CoNLL	NCBI	BC5CDR	Laptop	OntoNotes
# Instance	22,137	793	1,500	3,845	143,709
# Training	14,041	593	500	2,436	115,812
# Validation	3,250	100	500	609	5,000
# Test	3,453	100	500	800	22,897
# Entities	4	1	2	1	18
# LFs	16	5	9	3	17

observed by most LFs and chooses randomly from the tie if it exists. 2) **Snorkel** [21] is a context-free token-based simple graphical model which assumes the tokens are independent. 3) **HMM**, used in [16, 18, 25], is a popular weakly-supervised NER label model with certain context representation ability. 4) **Conditional HMM** (CHMM, 14) augments HMM by predicting the token-wise transition and emission probabilities from the BERT token embeddings through NNs. 5) **ConNet** [11] uses the context-aware attention mechanism to aggregate the CRF representations of different LFs.[3]

We also include 3 supervised methods as references: 1) a fully supervised **BERT-NER** model trained with human annotations, 2) the **best consensus** of LFs, which is an oracle that always selects the correct token annotations from the LFs; and 3) **CHMM-FE**, which is CHMM with the fixed ground-truth emissions as described in section 3.3.3. Every supervised method accesses the true labels in one way or another during training.

4.1.3 Evaluation Metrics. The NER label models are evaluated using the *micro*-averaged *entity-level* precision, recall, and F_1 scores. We calculate the metrics with the **seqeval** Python package.[4] The results come from the average of 5 random trials.

4.1.4 Implementation Details. Following Li et al. [14], we apply different BERT variants to different datasets. Specifically, we use bert-base-uncased [5] on CoNLL 2003, LaptopReview and OntoNotes 5.0, bioBERT [12] on NCBI-Disease and SciBERT [1] on BC5CDR.

In alignment with Zhang et al. [31], we evaluate Sparse-CHMM inductively. The model is trained on the training dataset and tested on the test dataset. The validation set is for early stopping and hyper-parameter fine-tuning. In addition, the WXOR scores are calculated on the combination of training and validation datasets. Please refer to appendix C for more implementation details and the model hyper-parameters that lead to the presented results.

4.2 Main Results

Table 3 shows the performance comparison of Sparse-CHMM and the baselines in the Wrench benchmark. Sparse-CHMM outperforms all other label models, achieving 3.01 average F_1 improvement over the strongest baselines. In general, the increment in recall contributes to most of Sparse-CHMM's performance gain. This is accredited to the well-founded emission structure, which prevents the correct annotations of inferior LFs from being neglected. On 3 out of 5 datasets, Sparse-CHMM has larger recall than the best consensus. This situation indicates that Sparse-CHMM is capable

[3]DWS [19] is similar to CHMM but has no open-source implementation.
[4]https://github.com/chakki-works/seqeval

Table 3: Evaluation results on the test datasets, presented as "F_1 (Precision / Recall)" in %.

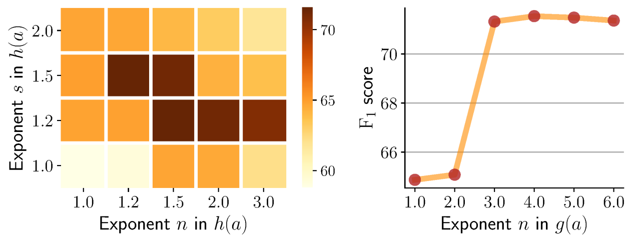
Models		CoNLL 2003	NCBI-Disease	BC5CDR	LaptopReview	OntoNotes 5.0
Supervised Methods	BERT-NER	90.74 (90.37 / 91.10)	88.89 (87.05 / 90.82)	88.81 (87.12 / 90.57)	81.34 (82.02 / 80.67)	84.11 (83.11 / 85.14)
	Best consensus	86.73 (98.62 / 77.39)	81.65 (99.85 / 69.06)	88.42 (99.86 / 79.33)	77.60 (100.0 / 63.40)	85.11 (97.35 / 75.61)
	CHMM-FE	71.43 (72.89 / 70.02)	81.86 (90.75 / 74.55)	86.45 (91.73 / 81.75)	72.38 (88.13 / 61.41)	67.99 (65.23 / 71.00)
Weakly Supervised Models	ConNet*	66.02 (67.98 / 64.19)	63.04 (74.55 / 55.16)	72.04 (77.71 / 67.18)	50.36 (63.04 / 42.73)	60.58 (59.43 / 61.83)
	MV*	60.36 (59.06 / 61.72)	78.44 (93.04 / 67.79)	80.73 (83.79 / 77.88)	73.27 (88.86 / 62.33)	58.85 (54.17 / 64.40)
	Snorkel*	62.43 (61.62 / 63.26)	78.44 (93.04 / 67.79)	83.50 (91.69 / 76.65)	73.27 (88.86 / 62.33)	61.85 (57.44 / 66.99)
	HMM*	62.18 (66.42 / 58.45)	66.80 (**96.79** / 51.00)	71.57 (**93.48** / 57.98)	73.63 (89.30 / 62.63)	55.67 (57.95 / 53.57)
	CHMM*	63.22 (61.93 / 64.56)	78.74 (93.21 / 68.15)	83.66 (91.76 / 76.87)	73.26 (88.79 / 62.36)	64.06 (59.70 / **69.09**)
	Sparse-CHMM	**71.53** (**73.80** / **69.39**)	**82.24** (93.18 / **73.60**)	**86.63** (89.56 / **83.88**)	**75.90** (**91.94** / **64.62**)	**64.85** (**61.26** / 68.88)

* Results are from the Wrench benchmark [31]. All weakly supervised models are evaluated with identical data and weak annotations.

Table 4: F_1 scores of ablation studies.

	CoNLL	NCBI	BC5CDR	Laptop	OntoNotes
Sparse-CHMM	**71.53**	**82.24**	**86.63**	**75.90**	64.85
Training Stages					
SpMM S1	69.73	77.89	86.15	73.59	64.15
SpMM S2	70.79	79.18	86.04	74.42	**64.92**
Model Components					
Naïve emiss	33.02	77.32	-*	71.77	-*
w/o h_{nsr}	58.66	-*	86.48	75.38	64.69
w/o SoftMax	62.86	80.61	82.56	73.71	53.19
w/o Dirichlet	64.70	81.34	86.05	73.51	64.64
w/o S2	71.18	81.48	86.02	73.11	63.90
Merge S2 & S3	71.27	79.81	85.95	71.49	64.22

* The model fails training; the output labels are all "O".
"SpMM" represents Sparse-CHMM and "S*i*" is short for "stage *i*".



(a) F_1 from $h_{n,s,r}$ with different n and s **(b) $g_{n,r}$ with different n**

Figure 6: Hyper-parameter studies on CoNLL 2003

of predicting entities observed by *no* LF, which is attributed to the well-trained token-wise transition probabilities.

Surprisingly, we find that Sparse-CHMM is generally superior to CHMM-FE, a model that incorporates true labels. The reason is that the ground-truth emission matrix in CHMM-FE is not necessarily the best when used as model parameters. Moreover, as the emission varies according to input sentences, using a fixed average value renounces the flexibility to accommodate the sentence pattern, which can be captured by Sparse-CHMM.

Nonetheless, a gap exists between Sparse-CHMM and the best consensus on CoNLL 2003 and OntoNotes 5.0. On these datasets, even CHMM-FE fails to achieve good performance. Apart from the difficulty these datasets bring with the increased number of entities and LFs, the reason is primarily the quality of their LFs (appendix B). The majority of LFs have low recall scores, letting the few reasonably-performed LFs dominate the training, which impedes the attention to the labels correctly observed by these LFs.

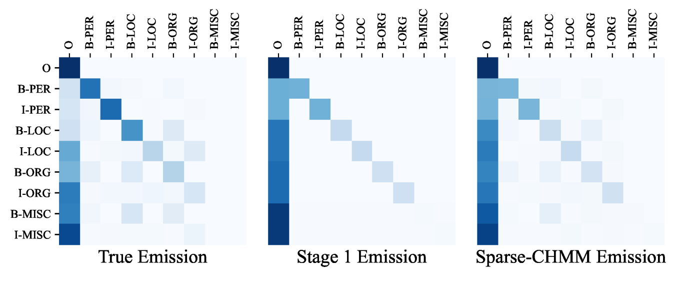
4.3 Ablation Studies

4.3.1 Training Stages. Table 4 shows the test performance from different training stages. The results demonstrate that the three-stage strategy contributes to Sparse-CHMM's performance in general. On BC5CDR, although stage 2 slightly weakens the model, the addon prior built in this stage is vital for stage 3's training. On OntoNotes 5.0, the minor decrease in stage 3 is because of the discrepancy between the training and test datasets, as we observe the valid F_1 increases from stage 2 to stage 3, as expected.

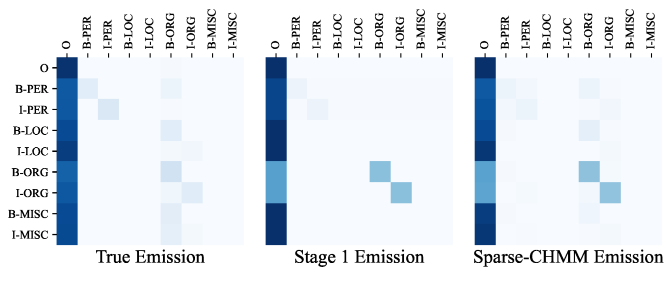
4.3.2 Model Components. To investigate the effectiveness of each model component, we include a series of ablation studies: 1) Sparse-CHMM with **naïve emission** matrices, which substitutes the reliability expansion function (4) by a Uniform over all non-diagonal values so that the emit-to-O probabilities are not emphasized as the original model; 2) Sparse-CHMM **w/o** the scaling function $h_{n,s,r}$, which is equal to substituting (2) by $\tilde{\boldsymbol{A}} = \hat{\boldsymbol{A}}$ or setting the exponents $n = s = 1$; 3) Sparse-CHMM **w/o** the **SoftMax** function defined in (1); 4) Sparse-CHMM **w/o** the **Dirichlet** sampling process during training; 5) Sparse-CHMM **w/o stage 2** training or the weighted XOR scores; 6) Sparse-CHMM that is trained with **merged stage 2 and 3**, *i.e.*, the transition parameters are not frozen in stage 2.

From Table 4, we conclude that each component introduced in section 3.2 contributes to the model performance. The model does not work with the naïve emission, which verifies the importance of the appropriately designed emission structure (3). We also find that both SoftMax and h are vital for Sparse-CHMM to predict good reliability scores. If Dirichlet sampling is deprecated, Sparse-CHMM has a higher chance to be trapped by the local optima. The 3-stage training strategy also makes the model optimization more stable.

4.3.3 Hyper-Parameters. Figure 6 presents the impact of different exponential terms of functions $h_{n,s,r}$ and $g_{n,r}$ to the model performance. The exponents help the element values of the base prior $\boldsymbol{\Lambda}$



(a) Emissions of LF BTC



(b) Emissions of LF crunchbase_uncased

Figure 7: Emission probabilities of two LFs on CoNLL 2003.

to rest within the desired range instead of leaving them arbitrary. This is essential for Sparse-CHMM to generate good base emission patterns and calculate accurate WXOR scores.

4.3.4 Model Efficiency. Table 5 supports that compared with CHMM, Sparse-CHMM is advanced in training efficiency. The advantage is more prominent with an increasing number of LFs and labels, aligning with our analysis in section 3.3.4. Since Sparse-CHMM is a simpler model than CHMM, it not only is faster to train per epoch, but also requires less epochs to be fully optimized. Thus, Sparse-CHMM is more efficient than CHMM from both perspectives.

Table 5: Training efficiency as "second/epoch (# epoch)[*]".

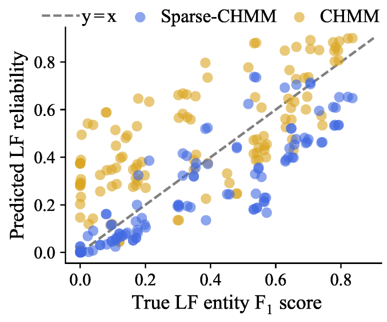
	CoNLL	NCBI	BC5CDR	Laptop	OntoNotes
CHMM	25.46 (50)	5.96 (55)	8.54 (72)	2.60 (56)	729.38 (28)
SpMM S1	9.64 (30)	4.66 (42)	4.02 (87)	2.06 (25)	277.60 (3)
SpMM S2	8.36 (5)	3.92 (3)	3.36 (5)	1.82 (6)	294.23 (2)
SpMM S3	8.50 (4)	4.04 (5)	3.54 (9)	1.84 (10)	250.16 (2)

[*]The number of epochs required to get similar results as presented in Table 3.
The batch size is ajusted such that the models consume similar computation resources during training and inference.
"SpMM" represents Sparse-CHMM and "S*i*" is short for "stage *i*".

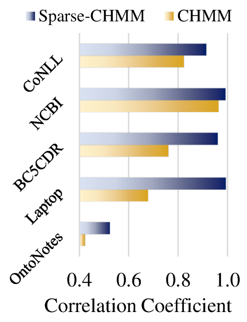
4.4 Case Studies

Figure 7 illustrates the averaged emission matrix from Sparse-CHMM and compares it with the ground-truth emissions. We can see that the model focuses on the diagonal and emit-to-O (*i.e.*, first-column) values in the first stage, and refines the emission matrix by adding the addon prior Δ to include the prominent off-diagonal values into the Dirichlet parameters. Comparing the diagonal values of the true and predicted emission matrices, we find that Sparse-CHMM fits the LF reliability scores well without using any clean labeled data. The reliability scores can also facilitate other tasks such as LF design and evaluation.

We further investigate the correlation between the predicted LF reliabilities and the entity F_1 scores of LFs. Figure 8a illustrates correlation, and Figure 8b measures the correlation quantitatively. We observe that the predicted reliabilities have a strong correlation with F_1 scores, reaching almost 1.0 correlation coefficients on 3 datasets. Compared with CHMM, Sparse-CHMM performs better



(a) Predicted LF reliabilities *vs.* LF entity F_1



(b) Correlation coefficients

Figure 8: Illustration of the correlation between the predicted LF reliability scores and the true LF F_1 scores. (a) shows the results from the CoNLL 2003 dataset.

in identifying unreliable LFs, which is critical for ruling out the incorrect observations.

5 RELATED WORKS

Weak supervision is a widely-investigated approach for natural language processing tasks such as sentence classification [21, 23, 24] and NER [11, 14–16, 18, 19, 25, 26]. It substitutes the laborious manual annotation process with multiple simple labeling functions, relieving the human effort to annotate large training data. However, the LFs are generally noisy and controversial, which makes their utilization challenging. Consequently, the primary track of weak supervision focuses on designing a robust label model that aggregates the weak annotations by de-noising them and resolving the conflict [11, 14–16, 18, 19, 21–26]. The hidden Markov model with multiple independent observation sequences is principled and popular for this task [14–16, 18, 19, 25]. Transferring from sequence classification to sequence labeling (*a.k.a.*, token classification), it extends the naïve Bayes generative model proposed by Ratner et al. [23] by incorporating the chronological dependency relationship through the transition matrix. The weak labels from different LFs are considered as multiple sets of observations that are conditionally independent given the latent variables. Among these works, Nguyen et al. [18] and Lison et al. [15, 16] use conventional HMMs. Safranchik et al. [25] design the linked HMM, which uses linking rules to model whether consecutive tokens belong to the same entity. Other works [14, 19] improve HMM's context representation with BERT embeddings and alleviate the Markov assumption.

In the data programming framework, people also train *end models*, which are deep supervised neural networks, with the label model outputs in seek of refining the results with the power of deep networks. One exception is the Consensus Network [11] that trains the label model and the end model jointly within a two-stage framework [31]. Another is ALT [14], which treats the end model as an additional LF and optimizes it alternately with the label model. Zhang et al. [31], however, show that the end model does not necessarily outperform the label model. Therefore, we focus on the label models here and leave end models to future works.

Our work is also related to the neuralized graphical models. Many efforts seek to inject neural networks into graphical models, especially for variational inference, such as the variational autoencoder (VAE) [10]. Other works include [28], which neuralizes the HMM with one observation set for the unsupervised part-of-speech tagging. Dai et al. [4] and Liu et al. [17] incorporate recurrent units into the hidden semi-Markov model to segment and label high-dimensional time series; Wiseman et al. [30] learn discrete template structures for conditional text generation also with neuralized graphical models. CHMM and DWS also fall into this category by predicting graphical models components through NNs.

6 CONCLUSION

We presented Sparse-CHMM, a label model that aggregates the weak annotations from multiple noisy NER labeling functions. In contrast to CHMM, Sparse-CHMM constructs the sentence-level sparse emission probabilities from the LF reliabilities predicted using the BERT sentence embeddings. The off-diagonal emission elements are further augmented by the weighted XOR scores, which estimate the possibilities of an LF observing incorrect entity labels. Wrapped in a three-stage training process with Dirichlet sampling, Sparse-CHMM outperforms all baseline label models on five representative datasets with different statistics and attributes. In addition, the approximated LF reliability scores strongly correlate with the true LF performance, making it eligible to contribute to other tasks such as automated LF generation and evaluation. In the future, we will consider leveraging prior heuristic knowledge to strengthen Sparse-CHMM's ability to distill clean NER labels from noisy LF annotations. In addition, we also plan to extend this technique to other sequence labeling tasks.

ACKNOWLEDGMENTS

This work was supported in part by ONR MURI N00014-17-1-2656, NSF IIS-2008334, IIS-2106961, CAREER IIS-2144338, and Kolon Industries.

REFERENCES

[1] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *EMNLP-IJCNLP*. 3615–3620.
[2] Benedikt Boecking, Willie Neiswanger, Eric P. Xing, and Artur Dubrawski. 2021. Interactive Weak Supervision: Learning Useful Heuristics for Data Labeling. In *ICLR*.
[3] Xiangning Chen, Ruochen Wang, Minhao Cheng, Xiaocheng Tang, and Cho-Jui Hsieh. 2021. DrNAS: Dirichlet Neural Architecture Search. In *ICLR*.
[4] Hanjun Dai, Bo Dai, Yan-Ming Zhang, Shuang Li, and Le Song. 2017. Recurrent Hidden Semi-Markov Model. In *ICLR*.
[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. 4171–4186.
[6] Rezarta Islamaj Dogan, Robert Leaman, and Zhiyong Lu. 2014. NCBI disease corpus: A resource for disease name recognition and concept normalization. *J. Biomed. Informatics* (2014), 1–10.
[7] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations. In *ACL*. 541–550.
[8] Martin Jankowiak and Fritz Obermeyer. 2018. Pathwise Derivatives Beyond the Reparameterization Trick. In *ICML*. 2240–2249.
[9] Edward Kim, Kevin Huang, Adam Saunders, Andrew McCallum, Gerbrand Ceder, and Elsa Olivetti. 2017. Materials Synthesis Insights from Scientific Literature via Text Extraction and Machine Learning. *Chemistry of Materials* (2017), 9436–9444.
[10] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.
[11] 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 *ACL*. 2134–2146.
[12] 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* (2019), 1234–1240.
[13] 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).
[14] Yinghao Li, Pranav Shetty, Lucas Liu, Chao Zhang, and Le Song. 2021. BERTifying the Hidden Markov Model for Multi-Source Weakly Supervised Named Entity Recognition. In *ACL-IJCNL*. 6178–6190.
[15] Pierre Lison, Jeremy Barnes, and Aliaksandr Hubin. 2021. skweak: Weak Supervision Made Easy for NLP. In *ACL-IJCNL*. 337–346.
[16] Pierre Lison, Jeremy Barnes, Aliaksandr Hubin, and Samia Touileb. 2020. Named Entity Recognition without Labelled Data: A Weak Supervision Approach. In *ACL*. 1518–1533.
[17] Hao Liu, Lirong He, Haoli Bai, Bo Dai, Kun Bai, and Zenglin Xu. 2018. Structured Inference for Recurrent Hidden Semi-Markov Model. In *IJCAI*. 2447–2453.
[18] An Thanh Nguyen, Byron Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and Predicting Sequence Labels from Crowd Annotations. In *ACL*. 299–309.
[19] Jerrod Parker and Shi Yu. 2021. Named Entity Recognition through Deep Representation Learning and Weak Supervision. In *ACL-IJCNLP Findings*. 3828–3839.
[20] Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 Task 4: Aspect Based Sentiment Analysis. In *SemEval*. 27–35.
[21] 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.* (2017), 269–282.
[22] Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2019. Training Complex Models with Multi-Task Weak Supervision. *AAAI* (2019), 4763–4771.
[23] Alexander Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data Programming: Creating Large Training Sets, Quickly. In *NIPS*. 3574–3582.
[24] Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising Multi-Source Weak Supervision for Neural Text Classification. In *EMNLP Findings*. 3739–3754.
[25] Esteban Safranchik, Shiying Luo, and Stephen H. Bach. 2020. Weakly Supervised Sequence Tagging from Noisy Rules. In *AAAI*. 5570–5578.
[26] Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. 2018. Learning Named Entity Tagger using Domain-Specific Dictionary. In *EMNLP*. 2054–2064.
[27] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *HLT-NAACL*. 142–147.
[28] Ke M. Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. 2016. Unsupervised Neural Hidden Markov Models. In *SPNLP*. 63–71.
[29] Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes Release 5.0 LDC2013T19.
[30] Sam Wiseman, Stuart Shieber, and Alexander Rush. 2018. Learning Neural Templates for Text Generation. In *EMNLP*. 3174–3187.
[31] Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. 2021. WRENCH: A Comprehensive Benchmark for Weak Supervision. *CoRR* (2021).

A TRAINING DETAILS

A.1 Training Objective

In this section, we focus on the computation of the expected complete data log likelihood Q defined in (8) as well as the training objective. The derivation largely aligns with [14], so we will skip some trivial steps and explanations.

By inserting (9) into (8) and specifying the expectation of z, we can write (8) as:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{i=1}^{L} p(z^{(0)} = i | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}) \log p(z^{(0)} = i) + \sum_{t=1}^{T}\sum_{i=1}^{L}\sum_{j=1}^{L} p(z^{(t-1)} = i, z^{(t)} = j | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}) \log \Psi_{i,j}^{(t)} + \sum_{t=1}^{T}\sum_{i=1}^{L} p(z^{(t)} = i | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}) \log \varphi_i^{(t)}, \tag{11}$$

where $\boldsymbol{\Psi}$ is the transition matrix and $\varphi_i^{(t)}$ is defined in (10). $p(z^{(0)})$ is the probability of the initial hidden state without any corresponding observations. As we can predict the token-wise transition matrix from the embeddings, we can simply set it to Uniform or, as Li et al. [14] proposed, set $p(z^{(0)} = 1)$ to 1 and $p(z^{(0)} = i), \forall i \in 2 : L$ to 0.

To calculate (11), we define the smoothed marginal $\boldsymbol{\gamma}^{(t)} \in [0, 1]^L$ as:

$$\gamma_i^{(t)} \triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}),$$

and the expected number of transitions $\boldsymbol{\xi}^{(t)} \in [0, 1]^{L \times L}$ as:

$$\xi_{i,j}^{(t)} \triangleq p(z^{(t-1)} = i, z^{(t)} = j | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}).$$

These two variables are acquired using the *forward-backward* algorithm.

First, we define the filtered marginal $\boldsymbol{\alpha} \in [0, 1]^L$ as:

$$\alpha_i^{(t)} \triangleq p(z^{(t)} = i | \boldsymbol{x}^{(1:t)}, \boldsymbol{e}^{(0:T)}),$$

and the conditional future evidence $\boldsymbol{\beta} \in [0, 1]^L$ as:

$$\beta_i^{(t)} \triangleq p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = i, \boldsymbol{e}^{(0:T)}).$$

In the forward pass, $\alpha_i^{(t)}$ is computed iteratively:

$$\begin{aligned}\alpha_i^{(t)} &\propto p(\boldsymbol{x}^{(t)} | z^{(t)} = i, \boldsymbol{e}^{(0)}) p(z^{(t)} = i | \boldsymbol{x}^{(1:t-1)}, \boldsymbol{e}^{(0:t)}) \\ &= \sum_{j=1}^{L} \varphi_i^{(t)} \Psi_{j,i}^{(t)} \alpha_j^{(t-1)},\end{aligned}$$

which can be written in the matrix form:

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

where $\odot$ is the element-wise product. We initialize $\boldsymbol{\alpha}$ with $\alpha_l^{(0)} = p(z^{(0)} = l), \forall l \in 1 : L$ since we have no observation at time step 0.

Similarly, we do the backward pass and compute $\boldsymbol{\beta}$:

$$\begin{aligned}\beta_i^{(t-1)} &= \sum_{j=1}^{L} p(z^{(t)} = j, \boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t+1:T)} | z^{(t-1)} = i, \boldsymbol{e}^{(0,t:T)}) \\ &= \sum_{j=1}^{L} \beta_j^{(t)} \varphi_j^{(t)} \Psi_{i,j}^{(t)}.\end{aligned}$$

In the matrix form, it becomes:

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

with base case:

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

With $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ calculated, $\gamma_i^{(t)}$ and $\xi_{i,j}^{(t)}$ can be written as:

$$\begin{aligned}\gamma_i^{(t)} &\propto p(z^{(t)} = i | \boldsymbol{x}^{(1:t)}, \boldsymbol{e}^{0:t}) p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = i, \boldsymbol{e}^{(0,t+1:T)}) \\ &= \alpha_i^{(t)} \beta_i^{(t)}, \\ \xi_{i,j}^{(t)} &\propto p(z^{(t-1)} = i | \boldsymbol{x}^{(1:t-1)} \boldsymbol{e}^{(0:t-1)}) p(\boldsymbol{x}^{(t)} | z^{(t)} = j, \boldsymbol{e}^{(0)}) \\ &\quad p(\boldsymbol{x}^{(t+1:T)} | z^{(t)} = j, \boldsymbol{e}^{(0,t+1:T)}) p(z^{(t)} = j | z^{(t-1)} = i, \boldsymbol{e}^{(t)}) \\ &= \alpha_i^{(t-1)} \varphi_j^{(t)} \beta_j^{(t)} \Psi_{i,j}^{(t)}.\end{aligned}$$

Written in the matrix form, they become:

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

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

Eventually, we insert $\boldsymbol{\gamma}$ and $\boldsymbol{\xi}$ into (11) to compute the value of Q. The training objective is to maximize Q, which can be readily done using the gradient ascend. Please refer to [14] for more details.

A.2 Inference

Same as Li et al. [14], we use the Viterbi algorithm to find the sequence of latent variables $\hat{z}^{(1:T)}$ that maximize the posterior:

$$\hat{\boldsymbol{z}}^{(1:T)} = \arg\max_{\boldsymbol{z}^{(1:T)}} p(\boldsymbol{z}^{(1:T)} | \boldsymbol{x}^{(1:T)}, \boldsymbol{e}^{(0:T)}).$$

This sequence of latent variables $\hat{z}^{(1:T)}$ is considered as Sparse-CHMM's approximation of the true labels $\boldsymbol{y}$.

B LABELING FUNCTION METRICS

Table 6–10 present the performance of each labeling function on the test set.

Table 6: LF performance on CoNLL 2003.

LF name	Precision	Recall	F_1
BTC	67.26	44.56	53.61
core_web_md	70.52	58.27	63.81
crunchbase_cased	37.76	6.69	11.37
crunchbase_uncased	32.75	7.31	11.96
full_name_detector	84.63	11.60	20.40
geo_cased	67.99	16.63	26.72
geo_uncased	64.35	20.20	30.75
misc_detector	85.26	20.68	33.29
multitoken_crunchbase_cased	72.73	3.40	6.50
multitoken_crunchbase_uncased	71.22	3.51	6.68
multitoken_geo_cased	72.06	1.74	3.39
multitoken_geo_uncased	66.83	2.35	4.55
multitoken_wiki_cased	94.36	16.01	27.37
multitoken_wiki_uncased	90.55	16.63	28.09
wiki_cased	75.62	35.48	48.30
wiki_uncased	71.50	38.95	50.43

Table 7: LF performance on NCBI-Disease.

LF name	Precision	Recall	F_1
tag-CoreDictionaryUncased	80.99	41.39	54.79
tag-CoreDictionaryExact	80.69	17.21	28.37
tag-CancerLike	34.88	1.58	3.03
tag-BodyTerms	68.52	3.91	7.39
link-ExtractedPhrase	96.88	36.11	52.62

Table 8: LF performance on BC5CDR.

LF name	Precision	Recall	F_1
tag-DictCore-Chemical	93.24	29.68	45.03
tag-DictCore-Chemical-Exact	89.55	3.26	6.29
tag-DictCore-Disease	84.19	26.91	40.78
tag-DictCore-Disease-Exact	81.40	1.08	2.13
tag-Organic Chemical	94.06	30.17	45.68
tag-Antibiotic	97.88	2.38	4.64
tag-Disease or Syndrome	79.01	11.81	20.55
link-PostHyphen	86.24	7.93	14.53
link-ExtractedPhrase	87.21	17.88	29.68

Table 9: LF performance on LaptopReview.

LF name	Precision	Recall	F_1
tag-CoreDictionary	72.63	51.61	60.34
link-ExtractedPhrase	97.46	29.40	45.18
link-ConsecutiveCapitals	35.29	0.92	1.79

Table 10: LF performance on OntoNotes 5.0.

LF name	Precision	Recall	F_1
Core_Keywords	51.25	8.01	13.86
Regex_Patterns	75.08	3.44	6.58
Numeric_Patterns	60.58	10.71	18.21
wiki_fine	77.24	53.15	62.97
money_detector	47.37	1.40	2.72
date_detector	66.74	4.34	8.15
number_detector	39.71	5.70	9.97
company_type_detector	65.14	1.43	2.80
full_name_detector	53.62	4.54	8.37
misc_detector	61.75	10.84	18.44
crunchbase_cased	23.07	4.80	7.94
crunchbase_uncased	22.63	4.84	7.97
geo_cased	62.87	9.03	15.79
geo_uncased	62.73	9.05	15.82
Multitoken_wiki	87.38	13.06	22.72
wiki_cased	48.61	15.62	23.64
wiki_uncased	48.24	15.68	23.67

C HYPER-PARAMETERS

The experiments are conducted on one GeForce RTX 2080 Ti GPU. The selected model hyper-parameters are presented in Table 11.

Table 11: Hyper-parameters.

		CoNLL	NCBI	BC5CDR	Laptop	OntoNotes
Training hyper-parameters						
Batch size		256	128	128	256	32
PT LR		5e-4	5e-4	5e-4	5e-4	1e-4
LR (S1)		2e-4	1e-3	1e-3	1e-4	1e-4
LR (S2)		4e-5	2e-4	2e-4	2e-5	2e-5
LR (S3)		2e-4	1e-3	1e-3	1e-4	1e-4
Use MV[1]		false	true	false	true	false
Model hyper-parameters						
Reliab LV		ENT	LB	ENT	LB	ENT
ν^{base}		10	2	2	2	2
ν^{expan}		1,000	1,500	1,500	1,000	1,000
h	n	1.2	1.2	0.9	0.8	1.1
	s	1.5	3	1.1	1.5	1
	r			$\frac{1}{K}$		
g	n			4		
	r (S1)	$\frac{1}{2L}$	$\frac{1}{20L}$	$\frac{1}{10L}$	$\frac{1}{20L}$	$\frac{1}{5L}$
	r (S2,3)	$\frac{1}{20L}$	$\frac{1}{20L}$	$\frac{1}{10L}$	$\frac{1}{20L}$	$\frac{1}{5L}$

"PT" is "pre-training"; "Reliab LV" is short for the "reliability level"; "LB" and "ENT" indicate "label"-level reliability (one score per label) and "entity"-level reliability (one score per entity).
"Si" represents "stage i"; paramters with no specified stage remain constant for all training stages.
L and K are the numbers of labels and LFs, respectively (section 2).
h is defined in (2); g is defined in (4).
[1] On some datasets, we add an additional majority voting LF to balance the annotations from existing LFs. The majority voting is only deployed in training but not involved in inference or evaluation.

To increase the training speed on the OntoNotes 5.0 dataset, the WXOR scores are calculated only on the validation set instead of the combination of training and validation sets.

D FUNCTION DESIGN CRITERIA

Here we introduce the design criteria for the reliability scaling function (2) and the reliability expansion function (4). There are infinite solutions that meet the criteria. We select the simplest polynomials for better interpretability and calculation efficiency.

D.1 Reliability Scaling Function

Function $f_{n,s,r}$, illustrated in Figure 3, is 1) continuous, smooth and monotonic; 2) passing through coordinates $(0,0)$ and $(1,1)$; and 3) having zero gradient at $(0,0)$ and $(1,1)$. It is designed such that its shape is controllable without making the function complicated.

D.2 Reliability Expansion Function

As shown in Figure 4, $g_{n,r}$ is 1) continuous, smooth and monotonic; 2) passing through $(0,1)$ and $(1,0)$; and 3) $\nabla_a g(a)|_{a=0} = \frac{1}{L-1}$. We want the emission from non-O latent state to non-O observation $\Lambda_{k,i,j\geq 2}$ to be close to Uniform when $\tilde{A}_{k,i}$ is small, which indicates that LF k may equally observe anything when the latent label is i. This builds the constraint of $\nabla_a \frac{1-a-g(a)}{L-2}|_{a=0} = \nabla_a a|_{a=0} = 1$, which leads to the third feature defined above. The hyperparameter r controls the threshold where we trust LF k enough to stop increasing the off-diagonal emissions.