Uncertainty-Aware Label Refinement for Sequence Labeling

Tao Gui[*12], **Jiacheng Ye**[*12], **Qi Zhang**[12],
Zhengyan Li[12], **Zichu Fei**[12], **Yeyun Gong**[3] **and Xuanjing Huang**[12]
[1]Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[2]School of Computer Science, Fudan University
[3]Microsoft Research Asia
{tgui16, yejc19, qz, zyl, zcfei19, xjhuang}@fudan.edu.cn
yegong@microsoft.com

Abstract

Conditional random fields (CRF) for label decoding has become ubiquitous in sequence labeling tasks. However, the local label dependencies and inefficient Viterbi decoding have always been a problem to be solved. In this work, we introduce a novel two-stage label decoding framework to model long-term label dependencies, while being much more computationally efficient. A base model first predicts draft labels, and then a novel two-stream self-attention model makes refinements on these draft predictions based on long-range label dependencies, which can achieve parallel decoding for a faster prediction. In addition, in order to mitigate the side effects of incorrect draft labels, Bayesian neural networks are used to indicate the labels with a high probability of being wrong, which can greatly assist in preventing error propagation. The experimental results on three sequence labeling benchmarks demonstrated that the proposed method not only outperformed the CRF-based methods but also greatly accelerated the inference process.

1 Introduction

Linguistic sequence labeling is one of the fundamental tasks in natural language processing. It has the goal of predicting a linguistic label for each word, including part-of-speech (POS) tagging, text chunking, and named entity recognition (NER). Benefiting from representation learning, neural network-based approaches can achieve state-of-the-art performance without massive handcrafted feature engineering (Ma and Hovy, 2016; Lample et al., 2016; Strubell et al., 2017; Peters et al., 2018; Devlin et al., 2019).

Although the use of representation learning to obtain better text representation is very successful, creating better models for label dependencies has always been the focus of sequence labeling tasks (Collobert et al., 2011; Ye and Ling, 2018; Zhang et al., 2018). Among them, the CRF layer integrated with neural encoders to capture label transition patterns (Zhou and Xu, 2015; Ma and Hovy, 2016) has become ubiquitous in sequence labeling tasks. However, CRF only captures the neighboring label dependencies and must rely on inefficient Viterbi decoding. Many of the recent methods try to introduce label embeddings to manage longer ranges of dependencies, such as two-stage label refinement (Krishnan and Manning, 2006; Cui and Zhang, 2019) and seq2seq (Vaswani et al., 2016; Zhang et al., 2018) frameworks. In particular, Cui and Zhang (2019) introduced a hierarchically-refined representation of marginal label distributions, which predicts a sequence of draft labels in advance and then uses the word-label interactions to refine them.

Although these methods can model longer label dependencies, they are vulnerable to error propagation: if a label is mistakenly predicted during inference, the error will be propagated and the other labels conditioned on this one will be impacted (Bengio et al., 2015). As shown in Figure 1, the label attention network (LAN) (Cui and

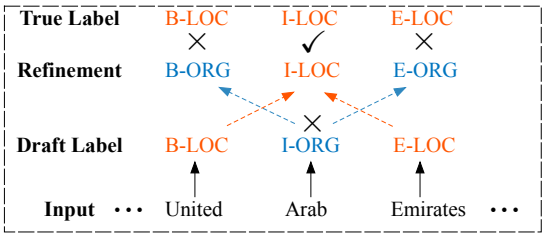

Figure 1: Schematic of label refinement framework (Cui and Zhang, 2019). The goal is refining the label of "Arab" using contextual labels and words, while the refinement of other correct labels may be negatively impacted by incorrect draft labels.

[*]Both authors contributed equally.

Draft	Uncertainty	Refinement	#Tokens
✔	0.018	✔ → ✘	**39**
✘	**0.524**	✘ → ✔	54

Table 1: Results of LAN with uncertainty estimation evaluated on CoNLL2003 test dataset. ✔ refers to the correct prediction, and ✘ refers to the wrong prediction. We use Bayesian neural networks (Kendall and Gal, 2017) to estimate the uncertainty. We can see that the uncertainty value of incorrect prediction is 29 times larger than that of correct predictions, which can effectively indicate the incorrect predictions.

Zhang, 2019) would negatively impact the correct predictions in the refinement stage. There are 39 correct tokens that have been incorrectly modified (Table 1). Hence, the model should selectively correct the labels with high probabilities of being incorrect, not all of them. Fortunately, we find that uncertainty values estimated by Bayesian neural networks (Kendall and Gal, 2017) can effectively indicate the labels that have a high probability of being incorrect. As shown in Table 1[1], the average uncertainty value of incorrect prediction is 29 times larger than that of correct predictions for the draft labels. Hence, we can easily set an uncertainty threshold to only refine the potentially incorrect labels and prevent side effects on the correct labels.

In this work, we propose a novel two-stage Uncertainty-Aware label refinement Network (UANet). At the first stage, the Bayesian neural networks take a sentence as input and yield all of the draft labels together with corresponding uncertainties. At the second stage, a two-stream self-attention model performs attention over label embeddings to explicitly model the label dependencies, as well as context vectors to model the context representations. All of these features are fused to refine the potentially incorrect draft labels. The above label refinement operations can be processed in parallel, which can avoid the use of Viterbi decoding of the CRF for a faster prediction. Experimental results on three sequence labeling benchmarks demonstrated that the proposed method not only outperformed the CRF-based methods but also significantly accelerated the inference process.

The main contributions of this paper can be summarized as follows: 1) we propose the use of Bayesian neural networks to estimate the uncertainty of predictions and indicate the potentially incorrect labels that should be refined; 2) we propose a novel two-stream self-attention refining framework to better model different ranges of label dependencies and word-label interactions; 3) the proposed parallel decoding process can greatly speed up the inference process; and 4) the experimental results across three sequence labeling datasets indicate that the proposed method outperforms the other label decoding methods.

[1]We slightly modified the code using Bayesian neural networks.

2 Related Work and Background

2.1 Sequence Labeling

Traditional sequence labeling models use statistical approaches such as Hidden Markov Models (HMM) and Conditional Random Fields (CRF) (Passos et al., 2014; Cuong et al., 2014; Luo et al., 2015) with handcrafted features and task-specific resources. With advances in deep learning, neural models could achieve competitive performances without massive handcrafted feature engineering (Chiu and Nichols, 2016; Santos and Zadrozny, 2014). In recent years, modeling label dependencies has been the other focus of sequence labeling tasks, such as using a CRF layer integrated with neural encoders to capture label transition patterns (Zhou and Xu, 2015; Ma and Hovy, 2016), and introducing label embeddings to manage longer ranges of dependencies (Vaswani et al., 2016; Zhang et al., 2018; Cui and Zhang, 2019). Our work is an extension of label embedding methods, which applies label dependencies and word-label interactions to only refine the labels with high probabilities of being incorrect. The probability of making a mistake is estimated using Bayesian neural networks, which will be described in the next subsection.

2.2 Bayesian Neural Networks

The predictive probabilities obtained by the softmax output are often erroneously interpreted as model confidence. However, a model can be uncertain in its predictions even with a high softmax output (Gal and Ghahramani, 2016a). Gal and Ghahramani (2016a) gives results showing that simply using predictive probabilities to estimate the uncertainty results in extrapolations with unjustified high confidence for points far from the training data. They verified that modeling a distribution over the parameters through Bayesian NNs can effectively reflect the uncertainty, and

Bernoulli Dropout is exactly one example of a regularization technique corresponding to an approximate variational distribution. Some typical examples of using Bernoulli distribution to estimate uncertainty are Bayesian CNN (Gal and Ghahramani, 2015) and variational RNN (Gal and Ghahramani, 2016b).

Given the dataset $\mathcal{D}$ with training inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and their corresponding outputs $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$, Bayesian inference looks for the posterior distribution of the parameters given the dataset $p(\mathbf{W}|\mathcal{D})$. This makes it possible to predict an output for a new input point $\mathbf{x}^*$ by marginalizing over all of the possible parameters, as follows:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) = \int p(\mathbf{y}^*|\mathbf{W}, \mathbf{x}^*)p(\mathbf{W}|\mathcal{D})\mathrm{d}\mathbf{W}. \tag{1}$$

Bayesian inference is intractable for many models because of the complex nonlinear structures and high dimension of the model parameters. Recent advances in variational inference introduced new techniques into the field. Among these, Monte Carlo Dropout (Gal and Ghahramani, 2016a) requires minimum modification to the original model. It is possible to use the variational inference approach to find an approximation $q^*_\theta(\mathbf{W})$ to the true posterior $p(\mathbf{W}|\mathcal{D})$ parameterized by a different set of weights θ, where the Kullback-Leibler (KL) divergence of the two distributions is minimized. The integral can be approximated as follows:

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) \approx \sum_{j=1}^{T} p(\mathbf{y}^*|\mathbf{W}_j, \mathbf{x}^*)q^*_\theta(\mathbf{W}_j). \tag{2}$$

In contrast to non-Bayesian networks, at test time, Dropouts are also activated. As a result, model uncertainty can be approximately evaluated by summarizing the variance of the model outputs from multiple forward passes.

3 Uncertainty-Aware Label Refinement

In this work, we propose a novel sequence labeling framework, which incorporates Bayesian neural networks to estimate the epistemic uncertainty of the draft labels. The uncertain labels that have a high probability of being wrong can be refined by a two-stream self-attention model using long-term label dependencies and word-label interactions. The proposed model is shown in Figure 2.

3.1 Variational LSTM for Uncertainty Estimation

Long short-term memory (LSTM) stands at the forefront of many recent developments in sequence labeling tasks. To facilitate comparison with LSTM-based models, variational LSTMs (Gal and Ghahramani, 2016b) as special Bayesian neural networks are used to encode sentences and determine the labels with a high probability of being wrong. Obviously, the uncertainty estimation methods can also be easily applied to other sequence labeling models, like the CNN and Transformer.

Word Representation Following Santos and Zadrozny (2014) and Lample et al. (2016), we use character information to enhance the word representation. Given a word sequence $S = \{w_1, w_2, \ldots, w_n\}$, the product of the one-hot encoded vector with an embedding matrix then gives a word embedding: $\mathbf{w}_i = \mathbf{e}^w(w_i)$, where $\mathbf{e}^w$ denotes a word embedding lookup table. Each word is made up of a sequence of characters $c_1, c_2, \ldots, c_l$. We adopt CNNs for character encoding and $\mathbf{x}^c_i$ denotes the output of character-level encoding. Then a word is represented by concatenating its word embedding and its character-level encoding: $\mathbf{x}_i = [\mathbf{w}_i; \mathbf{x}^c_i]$. All the word representations make up an embedding matrix $\mathbf{E} \in \mathsf{R}^{V \times D}$, where D is the embedding dimensionality of $\mathbf{x}$ and V is the number of words in the vocabulary.

Variational LSTM A common practice of Dropout technique on LSTM is that the technique should be used with the inputs and outputs of the LSTM alone. In contrast, the variational LSTM additionally applies Dropout on recurrent connections by repeating the same mask at each time step. Hence, the variational LSTM can model the uncertainty more accurately.

As shown in Figure 2, we use the same Dropout vectors $\mathbf{z}_x$ and $\mathbf{z}_h$ on four gates: "input", "forget", "output", and "input modulation" as follows:

$$\begin{aligned}
\begin{bmatrix}\mathbf{g}_i\\ \mathbf{i}_i\\ \mathbf{f}_i\\ \mathbf{o}_i\end{bmatrix} &= \left(\begin{bmatrix}\mathbf{W}^g\\ \mathbf{W}^i\\ \mathbf{W}^f\\ \mathbf{W}^o\end{bmatrix} \bullet \begin{bmatrix}\mathbf{x}'_i \odot \mathbf{z}_x\\ \mathbf{h}_{i-1} \odot \mathbf{z}_h\end{bmatrix} + \begin{bmatrix}\mathbf{b}^g\\ \mathbf{b}^i\\ \mathbf{b}^f\\ \mathbf{b}^o\end{bmatrix}\right)\\
\mathbf{c}_i &= \phi(\mathbf{g}_i) \odot \sigma(\mathbf{i}_i) + \mathbf{c}_{i-1} \odot \sigma(\mathbf{f}_i)\\
\mathbf{h}_i &= \sigma(\mathbf{o}_i) \odot \phi(\mathbf{c}_i),
\end{aligned} \tag{3}$$

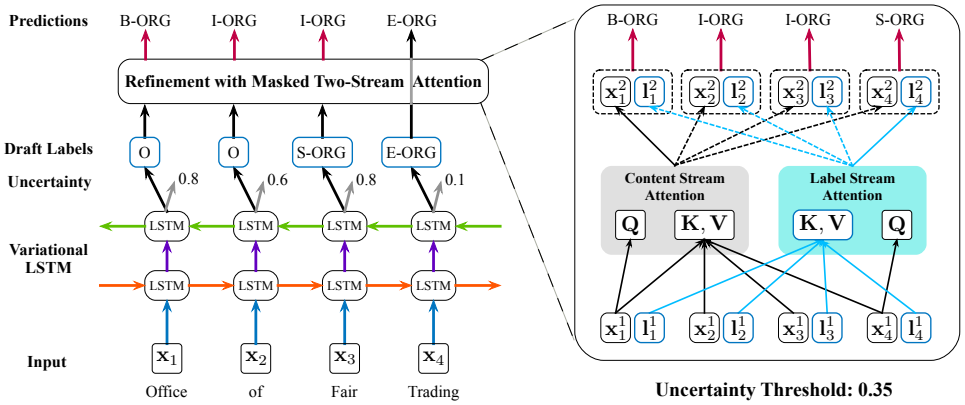

Figure 2: Graphical illustration of architecture and inference process for the proposed UANet. The variational LSTM outputs draft labels and model uncertainties simultaneously. The refinement only works on draft labels with threshold greater than 0.35.

where ϕ denotes the $\tanh$ function, and σ is the $\mathtt{sigmoid}$ function. $\odot$ and $\bullet$ represent the Hadamard product and matrix product, respectively. We assume that t is one of $\{g, i, f, o\}$. Then, $\theta = \{\mathbf{E}, \mathbf{W}^t\}$ and the Dropout rate r are the parameters of the variational LSTM.

Draft Labels and Uncertainty Estimation Assuming that we have completed the training and obtained the optimized approximated posterior $q^*_\theta(\mathbf{W})$ (the optimizing method is shown in § 3.3), at inference time, we can predict an output for a new input point by performing Monte Carlo integration in Eq.2 as follows:

$$\mathbf{p}_i(y = c|S, \mathcal{D}) \approx \frac{1}{M}\sum_{j=1}^{M} \mathtt{Softmax}(\mathbf{h}_i|\mathbf{W}_j) \quad (4)$$

with M sampled masked model weights $\mathbf{W}_j \sim q^*_\theta(\mathbf{W})$, where $q^*_\theta(\mathbf{W})$ is the Dropout distribution. In order to make the model with multiple sampling the same speed as the standard LSTM, we repeat the same input M times to form a batch and run in parallel on the GPU. Hence, M samples can be done concurrently in the forward passes, resulting in constant running time identical to that of standard Dropout (Gal and Ghahramani, 2016a), which is verified in Table 6.

Similar to classic sequence labeling models, the model applies $y^*_i = \mathtt{argmax}(\mathbf{p}_i)$ to obtain the draft label. Then the uncertainty of this probability vector $\mathbf{p}_i$ can be summarized using the entropy of the probability vector:

$$u_i = H(\mathbf{p}_i) = -\sum_{c=1}^{C} p_c \log p_c. \quad (5)$$

In this way, we can obtain the draft labels $Y^* = \{y^*_1, y^*_2, \ldots, y^*_n\}$ coupled with the corresponding epistemic uncertainties $U = \{u_1, u_2, \ldots, u_n\}$ for each input sentence. We find when the epistemic uncertainty u_i is larger than some threshold value , then the draft label y^*_i has a high probability of being wrong. Hence, we utilize a novel two-stream self-attention model to refine those uncertain labels using long-term label dependencies and word-label interactions.

3.2 Two-Stream Self-Attention for Label Refinement

Given the draft labels and corresponding epistemic uncertainties, we seek the help of label dependencies and word-label interactions to refine the uncertain labels. In order to refine the draft labels in parallel, we use the Transformer (Vaswani et al., 2017) incorporating relative position encoding (Dai et al., 2019) to model the words and draft labels.

In the standard Transformer, the attention score incorporating absolute position encoding between query q_i and key vector k_j can be decomposed as

$$\begin{aligned}\mathbf{A}^{abs}_{i,j} = &\mathbf{E}^\top_{x_i}\mathbf{W}^\top_q\mathbf{W}_k\mathbf{E}_{x_j} + \mathbf{E}^\top_{x_i}\mathbf{W}^\top_q\mathbf{W}_k\mathbf{U}_j \\ &+ \mathbf{U}^\top_i\mathbf{W}^\top_q\mathbf{W}_k\mathbf{E}_{x_j} + \mathbf{U}^\top_i\mathbf{W}^\top_q\mathbf{W}_k\mathbf{U}_j,\end{aligned} \quad (6)$$

where $\mathbf{U} \in \mathsf{R}^{L_{max}\times d}$ provides a set of positional encodings. The ith row $\mathbf{U}_i$ corresponds to the ith absolute position and L_{max} prescribes the maximum possible length to be modeled.

The relative position between labels is very important for modeling the label dependencies. Inspired by Dai et al. (2019), we modify the Eq.6 using the relative position encoding to model words and corresponding labels simultaneously,

but offer a different derivation, arriving at a new form of two-stream relative positional encodings. We not only provide a word-to-word interactions but also provide a word-to-label interactions correspondence to its counterpart. The relative position encodings are reparameterized as follows:

$$\begin{aligned}\mathbf{A}_{i,j}^{x2x} &= \mathbf{E}_{x_i}^{\top}\mathbf{W}_{qx}^{\top}\mathbf{W}_{kx}\mathbf{E}_{x_j} + \mathbf{E}_{x_i}^{\top}\mathbf{W}_{qx}^{\top}\mathbf{W}_{kR}\mathbf{R}_{i-j} \\ &\quad + \mathbf{u}_x^{\top}\mathbf{W}_{kx}\mathbf{E}_{x_j} + \mathbf{v}_x^{\top}\mathbf{W}_{kR}\mathbf{R}_{i-j} \\ \mathbf{A}_{i,m}^{x2l} &= \mathbf{E}_{x_i}^{\top}\mathbf{W}_{ql}^{\top}\mathbf{W}_{kl}\mathbf{E}_{y_m^*} + \mathbf{E}_{x_i}^{\top}\mathbf{W}_{ql}^{\top}\mathbf{W}_{kR}\mathbf{R}_{i-m} \\ &\quad + \mathbf{u}_l^{\top}\mathbf{W}_{kl}\mathbf{E}_{y_m^*} + \mathbf{v}_l^{\top}\mathbf{W}_{kR}\mathbf{R}_{i-m},\end{aligned} \tag{7}$$

where $\mathbf{A}_{i,j}^{x2x}$ and $\mathbf{A}_{i,m}^{x2l}$ denotes the attention from the ith word (x_i) to the jth word (x_j) and the ith word (x_i) to the mth label (y_m^*), respectively. $\mathbf{R}_{i-j}$ is the encoding of relative distance between position i and j, and $\mathbf{R}$ is the sinusoid matrix like Dai et al. (2019). $\varphi = \{\mathbf{W}, \mathbf{u}, \text{and } \mathbf{v}\}$ are learnable parameters.

Equipping the transformer with our proposed relative positional encoding, we finally arrive at the two-stream self-attention architecture. We summarize the computational procedure for one layer with a single attention head here:

$$\begin{aligned}\mathbf{V}_x &= \mathbf{E}_x\mathbf{W}_x, \mathbf{a}_x = \texttt{Softmax}(\mathbf{A}^{x2x})\mathbf{V}_x \\ \mathbf{V}_l &= \mathbf{E}_{y^*}\mathbf{W}_l, \mathbf{a}_l = \texttt{Softmax}(\mathbf{A}^{x2l})\mathbf{V}_l \\ \mathbf{o}_x &= \texttt{LayerNorm}(\texttt{Linear}(\mathbf{a}_x) + \mathbf{E}_x) \\ \mathbf{o}_l &= \texttt{LayerNorm}(\texttt{Linear}(\mathbf{a}_l) + \mathbf{E}_{y^*}) \\ \mathbf{H}_x &= \texttt{FeedForward}(\mathbf{o}_x) \\ \mathbf{H}_l &= \texttt{FeedForward}(\mathbf{o}_l).\end{aligned} \tag{8}$$

3.3 Training and Decoding

There are two networks to be optimized: one is variational LSTM for draft labels and uncertainty estimation, the other is two-stream self-attention model for label refinement. Our ultimate training goal is to minimize the total loss function on the two models: $\mathcal{L}_{total} = \mathcal{L}_1(\theta, r) + \mathcal{L}_2(\varphi)$.

The variational LSTM performs approximate variational inference. We use a simple Bernoulli distribution (Dropout) $q_\theta^*(\mathbf{W})$ in a tractable family to minimize the KL divergence to the true model posterior $p(\mathbf{W}|\mathcal{D})$. The minimization objective is given by (Jordan et al., 1999):

$$\mathcal{L}_1(\theta, r) = -\frac{1}{N}\sum_{i=1}^{N}\log p(y_i|\mathbf{W}_j) + \frac{1-r}{2N} \parallel \theta \parallel^2, \tag{9}$$

where N is the number of data points, and r is the Dropout probability to sample $\mathbf{W}_j \sim q_\theta^*(\mathbf{W})$.

For the two-stream self-attention model, we use the concatenation of $\mathbf{H}_x$ and $\mathbf{H}_l$ for the final prediction $\hat{y}_i = f(\mathbf{H}_x, \mathbf{H}_l|\mathbf{E}_x, \mathbf{E}_{y_m^*})$. In particular, we can optimize the model using cross entropy loss as:

$$\mathcal{L}_2(\varphi) = -\sum_{i=1}^{N} y_i \log \hat{y}_i, \tag{10}$$

where y_i is the one-hot vector of the label corresponding to w_i. When training is complete, we can obtain the draft labels $Y^* = \{y_1^*, y_2^*, \ldots, y_n^*\}$ and corresponding uncertainties $U = \{u_1, u_2, \ldots, u_n\}$ from variational LSTM, and refined labels $\hat{Y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n\}$ from two-stream self-attention model. To avoid the correct labels being incorrectly modified, we set an uncertainty threshold to distinguish which labels should be used, i.e., we use refined labels when $u_i >$ and vice versa (as an example, given $u_1 >$, $u_2 \leq$, and $u_n >$, decoding labels will become $\{\hat{y}_1, y_2^*, \ldots, \hat{y}_n\}$).

4 Experimental Setup

In this section, we describe the datasets across different sequence labeling tasks, including two English NER datasets and one POS tagging dataset. We also detail the baseline models for comparison. Finally, we clarify the hyperparameters configuration of our uncertainty-aware refinement network.

4.1 Datasets

We conduct experiments on three sequence labeling datasets. The statistics are listed in Table 2.

CoNLL2003. The shared task of CoNLL2003 dataset (Tjong Kim Sang and De Meulder, 2003) for named entity recognition is collected from Reuters Corpus. The dataset divide name entities into four different types: persons, locations, organizations, and miscellaneous entities. We use the BIOES tag scheme instead of standard BIO2, which is the same as Ma and Hovy (2016).

OntoNotes 5.0. English NER dataset OntoNotes 5.0 (Weischedel et al., 2013) is a large corpus consists of various genres, such as newswire, broadcast, and telephone speech. Named entities are labeled in eleven types and values are specifically divided into seven types, like DATE, TIME, ORDINAL.

WSJ. Wall Street Journal portion of Penn Treebank (Marcus et al., 1993), which contains 45 types of part-of-speech tags. We adopts standard splits following previous works (Collins, 2002; Manning,

Dataset	#Train	#Dev	#Test	class
CoNLL2003	204,567	51,578	46,666	17
OntoNotes	1,088,503	147,724	152,728	73
WSJ	912,344	131,768	129,654	45

Table 2: Statistics of CoNLL2003, OntoNotes and WSJ datasets, where # represents the number of tokens in datasets. The class number of NER datasets is counted under BIOES tag scheme.

2011), selecting section 0-18 for training, section 19-21 for validation and section 22-24 for test.

4.2 Compared Methods

In this work, we mainly focus on improving decoding efficiency and enhancing label dependencies. Thus, we make comparisons with the classic methods that have different decoding layers, such as Softmax, CRF, and LAN frameworks. We also compare some recent competitive methods, such as Transformer, IntNet (Xin et al., 2018), and BERT (Devlin et al., 2019).

BiLSTM-Softmax. This baseline uses bidirectional LSTM to reprensent a sequence. The BiLSTM concatenates the forward hidden state $\overrightarrow{\mathbf{h}}_i$ and backward hidden state $\overleftarrow{\mathbf{h}}_i$ to form an integral representation $\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]$. Finally, sentence representation $\mathbf{H} = \{\mathbf{h}_i, \cdots, \mathbf{h}_n\}$ is fed to softmax layer for predicting.

BiLSTM-CRF. A CRF layer is used on top of the hidden vectors $\mathbf{H}$ (Ma and Hovy, 2016). The CRF can model bigram interactions between two successive labels (Lample et al., 2016) instead of making independent labeling decisions for each output. In the decoding time, the Viterbi algorithm is used to find the highest scored label sequence over an input word sequence.

BiLSTM-Seq2seq. To model longer label dependencies, Zhang et al. (2018) predicts a sequence of labels as a sequence to sequence problem.

BiLSTM-LAN. The label attention network (LAN) (Cui and Zhang, 2019) introduces label embedding, and uses consecutive attention layers on the label embeddings to refine the draft labels. It achieves the state-of-the-art results on several sequence labeling tasks.

Rel-Transformer. This baseline model adopts self-attention mechanism with relative position representations (Vaswani et al., 2017; Dai et al., 2019).

Models	CoNLL2003	OntoNotes	WSJ
Chiu and Nichols (2016)	90.91	86.28	-
Strubell et al. (2017)	90.54	86.84	-
Liu et al. (2018)	91.24	-	97.53
Chen et al. (2019)	91.44	87.67	-
BiLSTM-CRF (Ma and Hovy, 2016)	91.21	86.99	97.51
BiLSTM-Softmax (Yang et al., 2018)	90.77	83.76	97.51
BiLSTM-Seq2seq (Zhang et al., 2018)	91.22	-	97.59
Rel-Transformer (Dai et al., 2019)	90.70	87.45	97.49
BiLSTM-LAN (Cui and Zhang, 2019)	90.77*	88.16	97.58
BiLSTM-UANet (M = 8)	**91.60**	**88.39**	**97.62**

Table 3: Main results on three sequence labeling datasets. * indicates the results by running Cui and Zhang (2019)'s released code[5].

4.3 Hyper-parameter Settings

Following (Ma and Hovy, 2016), we use the same 100-dimensional GloVe embeddings[2] as initialization. We use 1-layer variational LSTM with a hidden size of 400 to create draft labels. The vanilla dropout after the embedding layer and the variational dropout is set to 0.5 and 0.25, respectively. We use 2 layers of multi-head transformer for WSJ and CoNLL2003 and 3 for OntoNotes dataset to refine the label. The number of heads is chosen from $\{5, 7, 9\}$, and the dimension of each head is chosen from $\{80, 120, 160\}$ via grid search. We use SGD as the optimizer for variational LSTM and Adam (Kingma and Ba, 2014) for transformer. Learning rates are set to 0.015 for SGD on CoNLL2003 and Ontonotes datasets and 0.2 on WSJ dataset. The learning rates for Adam are set to 0.0001 for all datasets. F1 score and accuracy are used for NER and POS tagging, respectively. All experiments are implemented in NCRF++ (Yang and Zhang, 2018) and conducted using a GeForce GTX 1080Ti with 11GB memory. More details are shown in our codes[3].

5 Results and Analysis

In this section, we present the experimental results of the proposed and baseline models. We show that the proposed method not only achieves better performance but also has a significant speed advantage. Since our contribution is mainly focused on the label decoding layer, the proposed model can also be combined with the latest pre-trained model to further improve performance.

[2]http://nlp.stanford.edu/projects/glove/
[3]https://github.com/jiacheng-ye/UANet
[4]https://github.com/Nealcly/BiLSTM-LAN

Models	CoNLL2003	OntoNotes	WSJ
BiLSTM-UANet	**91.60**	**88.39**	**97.62**
- Label information	91.23	87.84	97.57
- Variational LSTM			
Rel-Transformer-Softmax	90.70	87.45	97.49
Rel-Transformer-CRF	91.22	87.77	97.56
- Two-stream self-attention			
Variational LSTM-Softmax	90.83	87.11	97.46
Variational LSTM-CRF	91.20	87.63	97.55

Table 4: Ablation study of UANet.

Models	F_1
IntNet-BiLSTM-Softmax (Xin et al., 2018)	91.43
IntNet-BiLSTM-CRF	91.64
IntNet-UANet	**91.80**
BERT-Softmax (Devlin et al., 2019)	91.62
BERT-CRF	91.71
BERT-UANet	**92.02**

Table 5: Results on CoNLL2003 test set. We implement BERT for NER task without document-level information. Original result of BERT in (Devlin et al., 2019) was not achieved with the current version of the library. See a discussion in (Stanislawek et al., 2019) and the reported results at (Zhang et al., 2019).

	CoNLL2003	OntoNotes	WSJ
Average Sentence Length	13	18	24
BiLSTM-Softmax	3,443	2,910	3,767
BiLSTM-CRF	1,433	950	801
BiLSTM-LAN	949	773	943
BiLSTM-Seq2seq	1,084	842	751
BiLSTM-UANet (M = 1)	1,630	1,262	1,192
BiLSTM-UANet (M = 8)	1,474	1,129	1,044
BERT-CRF	254	231	189
BERT-UANet (M = 8)	335	266	214

Table 6: Comparison of inference speed. M represents for the number of sampling. We show how many sentences the model can process per second.

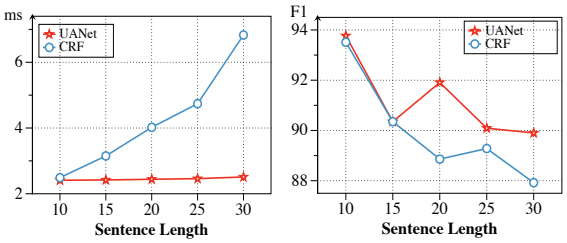

Figure 3: Speed and F1 against sentence length.

5.1 Main Results

Table 3 reports model performances on CoNLL2003, OntoNotes, and WSJ dataset, which shows that the proposed method not only can achieve state-of-the-art results on NER task but also is effective on other sequence labeling tasks, like POS tagging. The previous methods leverage rich handcrafted features (Huang et al., 2015; Chiu and Nichols, 2016), CRF decoding (Strubell et al., 2017), and longer range label dependencies (Zhang et al., 2018; Cui and Zhang, 2019). Compared with these methods, our UANet model gives better results. Benefitting from the strong capability of modeling long-term label dependencies, the UANet outperforms models with the CRF inference layer by a large margin. Moreover, different from the seq2seq and LAN models that also leverage label dependencies, our UANet model integrates model uncertainty into the refinement stage to avoid side effects on correct draft labels. As a result, it outperforms LAN and seq2seq models on all of the three datasets.

5.2 Ablation Study

To study the contribution of each component in BiLSTM-UANet, we conducted ablation experiments on the three datasets and display the results in Table 4. The results show that the model's performance is degraded if the draft label information is removed, indicating that label dependencies are useful in the refinement. We also find that both the variational LSTM and two-stream self-attention play an important role in label refinement. Even though we replace any component with the CRF layer, the performance will be seriously hurt.

We also give our model more complex character representations (IntNet) or use the pretrained model (BERT) to replace the Glove embeddings. We fine-tune the BERT for each task. The results are shown in Table 5. We find that the contribution of our model and more complex word representations may be orthogonal, i.e., whether or not the UANet uses the IntNet and BERT, our methods have similar improvements, because of better modeling label dependencies.

5.3 Efficient Advantage

Table 6 shows a comparison of inference speeds. BiLSTM-UANet processes 1,630, 1,262, and 1,192 sentences per second on the CoNLL2003, OntoNotes, and WSJ development data, respectively, outperforming BiLSTM-CRF by 13.7%, 32.8% and 48.8%, respectively. We can see that for the dataset with a longer average length, the speed of inference will be more advantageous. Because the model calculates uncertainties through parallel sampling the same input multiple times, the inference time of the BiLSTM-UANet (M = 8)

Text	...	striker	Viorel	Ion	of	Otelul	Galati	and	defender	Liviu	Ciobotariu	of	National	Bucharest	...	University	of	Yangon	...
BiLSTM-CRF	...	O	B-PER	E-PER	O	**B-PER**	**E-PER**	O	O	B-PER	E-PER	O	**B-LOC**	**E-LOC**	...	...	**O**	**O**	**S-LOC**	...
Draft Label	...	O	B-PER	E-PER	O	*B-ORG*	*E-ORG*	O	O	B-PER	E-PER	O	*B-ORG*	*E-ORG*	...	...	*B-ORG*	*I-ORG*	**E-LOC**	...
Refinement	...	O	B-PER	E-PER	O	*B-ORG*	*E-ORG*	O	O	B-PER	E-PER	O	*B-ORG*	*E-ORG*	...	...	**B-LOC**	*I-ORG*	*E-ORG*	...
Uncertainty	...	0.001	0.005	0.047	0.004	*0.532*	*0.605*	0.000	0.000	0.001	0.014	0.001	*0.818*	*0.927*	...	...	0.302	*0.816*	*0.800*	...
Final Prediction	...	O	B-PER	E-PER	O	B-ORG	E-ORG	O	O	B-PER	E-PER	O	B-ORG	E-ORG	...	...	B-ORG	I-ORG	E-ORG	...

Table 7: NER cases analysis. Contents with **bold red** and *italic blue* styles represent incorrect and correct entities, respectively. Draft labels with uncertainty greater than 0.35 will be refined.

only slightly increases.

To further investigate the influence of the different sentence lengths, we analyze the inference speed of the UANet and CRF on the CoNLL2003 development set, which is split into five parts according to sentence length. We ruled out the influence of the text encoder and only counted the time of label decoding. The left subfigure in Figure 3 shows the decoding speed on the different sentence lengths. The results reveal that as the sentence length increases, the speed of the UANet is relatively stable, while the speed of the CRF decreases substantially. Due to the UANet's parallelism, when processing the sentence longer than 30, the UANet is nearly 3 times faster than the CRF. In addition, we exhibit the F1 score of the sentences with different lengths in right subfigure. It is worth noting that the UANet outperforms the CRF by a large margin when the length of the sentence is greater than 15, verifying the UANet's superiority in long-term label dependencies.

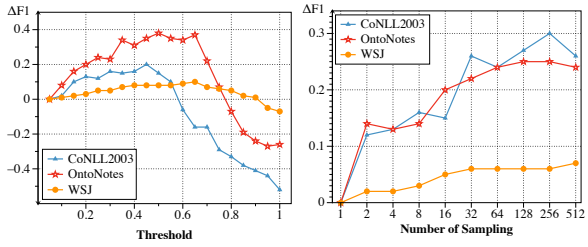

Figure 4: F1 variation under different uncertainty thresholds and numbers of sampling in variational LSTM, respectively. The results are evaluated on the development sets. F1 represents the F1 scores at different steps minus the initial results.

5.4 Discussion

Uncertainty Threshold. In order to investigate the influence of uncertainty threshold , we evaluate the performance with different uncertainty thresholds on three datasets, as shown in Figure 4. = 0 represents that the model uses all of the refined labels as final predictions. As the threshold gets larger, the performance of UANet can improve by reducing the negative effects on correct draft labels. However, when is too large, the model mainly uses draft labels as final predictions, resulting in performance degradation, which verifies our motivation that a reasonable uncertainty threshold can avoid side effects on correct draft labels.

Number of Sampling. We also investigate the influence of the number of sampling in the variational LSTM as shown in Figure 4. The results meet our expectation that a larger number of sampling can lead to better performance because a larger number of sampling can make the model better approximate the posterior $p(\mathbf{W}|\mathcal{D})$.

Case Study. Table 7 shows two cases from CoNLL2003 NER dataset. The first case reflects the necessity of modeling higher-order dependencies in the NER task. UANet can learn the label consistency of two phrases near the word `and`. Moreover, seq2seq decoding model (Zhang et al., 2018) refines the labels in a left-to-right way and can't refine the previous labels in this case. The second case shows the effectiveness of the uncertainty threshold in mitigating the side effect of incorrect refinement. In this case, the refinement model is affected by the incorrect label of `Yangon` (E-LOC) when predicting the word `University`. Since the uncertainty value of `University` is lower than the threshold, our model can get the correct results.

6 Conclusions

In this work, we introduce a novel sequence labeling framework that incorporates Bayesian neural networks to estimate model uncertainty. We find that the model uncertainty can effectively indicate the labels with a high probability of being wrong. The proposed method can selectively refine the uncertain labels to avoid the side effects of the refinement on correct labels. In addition, the proposed model can capture different ranges of label dependencies and word-label interactions in parallel, which can avoid the use of Viterbi

decoding of the CRF for a faster prediction. Experimental results across three sequence labeling datasets demonstrated that the proposed method significantly outperforms the previous methods.

Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No. 2018YFC0831105, 2018YFB1005104, 2017YFB1002104), National Natural Science Foundation of China (No. 61751201, 61976056, 61532011), Shanghai Municipal Science and Technology Major Project (No.2018SHZDZX01), Science and Technology Commission of Shanghai Municipality Grant (No.18DZ1201000, 17JC1420200).

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179.

Hui Chen, Zijia Lin, Guiguang Ding, Jian-Guang Lou, Yusen Zhang, and Börje F. Karlsson. 2019. Grn: Gated relation network to enhance convolutional neural network for named entity recognition. In *AAAI*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537.

Leyang Cui and Yue Zhang. 2019. Hierarchically-refined label attention network for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4106–4119.

Nguyen Viet Cuong, Nan Ye, Wee Sun Lee, and Hai Leong Chieu. 2014. Conditional random field with high-order dependencies for sequence labeling and segmentation. *The Journal of Machine Learning Research*, 15(1):981–1009.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

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

Yarin Gal and Zoubin Ghahramani. 2015. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*.

Yarin Gal and Zoubin Ghahramani. 2016a. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

Yarin Gal and Zoubin Ghahramani. 2016b. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584.

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

Vijay Krishnan and Christopher D Manning. 2006. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1121–1128. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*, pages 260–270.

Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Fangzheng Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 879–888.

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

Christopher D. Manning. 2011. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part I*, CICLing'11, pages 171–189, Berlin, Heidelberg. Springer-Verlag.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 78–86.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826.

Tomasz Stanislawek, Anna Wróblewska, Alicja Wójcicka, Daniel Ziembicki, and Przemyslaw Biecek. 2019. Named entity recognition-is there a glass ceiling? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 624–633.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2670–2680.

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.

Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. 2016. Supertagging with lstms. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 232–237.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Yingwei Xin, Ethan Hart, Vibhuti Mahajan, and Jean-David Ruvini. 2018. Learning better internal structure of words for sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2584–2593, Brussels, Belgium. Association for Computational Linguistics.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jie Yang and Yue Zhang. 2018. NCRF++: An open-source neural sequence labeling toolkit. In *Proceedings of ACL 2018, System Demonstrations*, pages 74–79, Melbourne, Australia. Association for Computational Linguistics.

Zhixiu Ye and Zhen-Hua Ling. 2018. Hybrid semi-markov crf for neural sequence labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 235–240.

Yuan Zhang, Hongshen Chen, Yihong Zhao, Qun Liu, and Dawei Yin. 2018. Learning tag dependencies for sequence tagging. In *IJCAI*, pages 4581–4587.

Zhuosheng Zhang, Bingjie Tang, Zuchao Li, and Hai Zhao. 2019. Modeling named entity embedding distribution into hypersphere. *arXiv preprint arXiv:1909.01065*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual*

Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1127–1137.