SEMI-SUPERVISED NAMED ENTITY RECOGNITION WITH CRF-VAES

Anonymous authors
Paper under double-blind review

ABSTRACT

We investigate methods for semi-supervised learning (SSL) of a neural linear-chain conditional random field (CRF) for Named Entity Recognition (NER) by treating the tagger as the amortized variational posterior in a generative model of text given tags. We first illustrate how to incorporate a CRF in a VAE, enabling end-to-end training on semi-supervised data. We then investigate a series of increasingly complex deep generative models of tokens given tags enabled by end-to-end optimization, comparing the proposed models against supervised and strong CRF SSL baselines on the Ontonotes5 NER dataset. We find that our best proposed model consistently improves performance by $\approx 1\%$ F1 in low- and moderate-resource regimes and easily addresses degenerate model behavior in a more difficult, partially supervised setting.

1 INTRODUCTION

Named entity recognition (NER) is a critical subtask of many domain-specific natural language understanding tasks in NLP, such as information extraction, entity linking, semantic parsing, and question answering. State-of-the-art models treat NER as a tagging problem (Lample et al., 2016; Ma & Hovy, 2016; Strubell et al., 2017; Akbik et al., 2018), and while they have become quite accurate on benchmark datasets in recent years (Lample et al., 2016; Ma & Hovy, 2016; Strubell et al., 2017; Akbik et al., 2018; Peters et al., 2018; Devlin et al., 2018), utilizing them for new tasks is still expensive, requiring a large corpus of exhaustively annotated sentences (Snow et al., 2008). This problem has been largely addressed by extensive pretraining of high-capacity sentence encoders on massive-scale language modeling tasks (Peters et al., 2018; Devlin et al., 2018; Howard & Ruder, 2018; Radford et al., 2019; Liu et al., 2019b), but it is natural to ask if we can squeeze more signal from our unlabeled data.

Latent-variable generative models of sentences are a natural approach to this problem: by treating the tags for unlabeled data as latent variables, we can appeal to the principle of maximum marginal likelihood (Berger, 1985; Bishop, 2006) and learn a generative model on both labeled and unlabeled data. For models of practical interest, however, this presents multiple challenges: learning and prediction both require an intractable marginalization over the latent variables and the specification of the generative model can imply a posterior family that may not be as performant as the current state-of-the-art discriminative models.

We address these challenges using a semi-supervised Variational Autoencoder (VAE) (Kingma et al., 2014), treating a neural tagging CRF as the approximate posterior. We address the issue of optimization through discrete latent tag sequences by utilizing a differentiable relaxation of the Perturb-and-MAP algorithm (Papandreou & Yuille, 2011; Mensch & Blondel, 2018; Corro & Titov, 2018), allowing for end-to-end optimization via backpropagation (Rumelhart et al., 1988) and SGD (Robbins & Monro, 1951). Armed with this learning approach, we no longer need to restrict the generative model family (as in Ammar et al. (2014); Zhang et al. (2017)), and explore the use of rich deep generative models of text given tag sequences for improving NER performance. We also demonstrate how to use the VAE framework to learn in a realistic annotation scenario where we only observe a biased subset of the named entity tags.

Our contributions can be summarized as follows:

1. We address the problem of semi-supervised learning (SSL) for NER by treating a neural CRF as the amortized approximate posterior in a discrete structured VAE. To the best of our knowledge, we are the first to utilize VAEs for NER.
2. We explore several variants of increasingly complex deep generative models of text given tags with the goal of improving tagging performance. We find that a joint tag-encoding Transformer (Vaswani et al., 2017) architecture leads to an $\approx 1\%$ improvement in F1 score over supervised and strong CRF SSL baselines.
3. We demonstrate that the proposed approach elegantly corrects for degenerate model performance in a more difficult *partially supervised* regime where sentences are not exhaustively annotated and again find improved performance.
4. Finally, we show the utility of our method in realistic low- and high-resource scenarios, varying the amount of unlabeled data. The resulting high-resource model is competitive with state-of-the-art results and, to the best of our knowledge, achieves the highest reported F1 score (88.4%) for models that do not use additional labeled data or gazetteers.

2 METHODS

We first introduce the tagging problem and tagging model. We then detail our proposed modeling framework and architectures.

2.1 PROBLEM STATEMENT

NER is the task of assigning coarsely-typed categories to contiguous spans of text. State-of-the-art approaches (Lample et al., 2016; Ma & Hovy, 2016; Strubell et al., 2017; Akbik et al., 2018; Liu et al., 2019a) do so by treating span extraction as a tagging problem, which we now formally define.

We are given a tokenized text sequence $x_{1:N} \in \mathcal{X}^N$ and would like to predict the corresponding tag sequence $y_{1:N} \in \mathcal{Y}^N$ which correctly encodes the observed token spans. [1] In this work, we use the BILOU (Ratinov & Roth, 2009) tag-span encoding, which assigns four tags for each of the C span categories (e.g., `B-PER, I-PER, L-PER, U-PER` for the `PERSON` category.) The tag types `B, I, L, U` respectively encode beginning, inside, last, and unary tag positions in the original span. Additionally we have one `O` tag for tokens that are not in any named entity span. Thus our tag space has size $|\mathcal{Y}| = 4C + 1$.

2.2 TAGGING CRF

We call the NER task of predicting tags for tokens *inference*, and model it with a discriminative distribution $q_\phi(y_{1:N}|x_{1:N})$ having parameters ϕ. Following state-of-the-art NER approaches (Lample et al., 2016; Ma & Hovy, 2016; Strubell et al., 2017; Akbik et al., 2018), we use a neural encoding of the input followed by a linear-chain CRF (Lafferty et al., 2001) decoding layer on top.

We use the same architecture for q_ϕ throughout this work, as follows:

1. Encode the token sequence, represented as byte-pairs, with a fixed pretrained language model. [2] That is, we first calculate:

$$h^0_{1:N} = \text{Pretrained-LM}(x_{1:N}), \;\; h^0_{1:N} \in \mathbb{R}^{N \times d_{\text{LM}}}$$

   In our first experiments exploring the use of pretrained autoregressive information for generation (§3.1), we use the GPT2-SM model (Radford et al., 2019; Hugging Face, 2019). In the experiments after (§3.2) we use the RoBERTa-LG model (Liu et al., 2019b; Hugging Face, 2019).

2. Down-project the states: $h^1_{1:N} = h^0_{1:N} W_1 + b_1, \;\; W_1 \in \mathbb{R}^{d_{\text{LM}} \times d_{y_q}}, \; b_1 \in \mathbb{R}^{d_{y_q}}$

[1] We will often omit sequence boundaries ($x \leftarrow x_{1:N}$) to save space, but will always index individual elements x_i.

[2] We represent the tags at the byte-pair level to ensure alignment between the number of tokens and tags for the generative models in §2.3

3. Compute local tag scores: $\mathrm{s}_{y_i} = v_y^\top h_i^1 + b_{2,y},\ \ v_y \in \mathbb{R}^{d_{y_q}},\ b_2 \in \mathbb{R}^{|\mathcal{Y}|}$
4. Combine local and transition potentials: $\psi_{y_i,y_{i+1}} = s_{y_i} + T_{y_i,y_{i+1}},\ \ T_{y_i,y_{i+1}} \in \mathbb{R}$
5. Using special start and end states $y_0 = *, y_{N+1} = \diamond$ with binary potentials $\psi_{*,y} = T_{*,y}, \psi_{y,\diamond} = T_{y,\diamond}$ and the forward algorithm (Lafferty et al., 2001) to compute the the partition function Z, we can compute the joint distribution:

$$q_\phi(y_{1:N}|x_{1:N}) = \exp\{\sum_{i=0}^{N} \psi_{y_i,y_{i+1}} - \log Z(\psi)\} \tag{1}$$

Our tagging CRF has trainable parameters $\phi = \{W_1, b_1, V, b_2, T\}$ [3] and we learn them on a dataset of fully annotated sentences $\mathcal{D}_S = \{(x^i_{1:N^i}, y^i_{1:N^i})\}$ using stochastic gradient descent (SGD) and maximum likelihood estimation.

$$\mathcal{L}^q_S(\phi; \mathcal{D}_S) = \sum_{(x,y)\in\mathcal{D}_S} \log q_\phi(y|x) \tag{2}$$

2.3 Semi-Supervised CRF-VAE

We now present the CRF-VAE, which treats the tagging CRF as the amortized approximate posterior in a Variational Autoencoder. We first describe our loss formulations for semi-supervised and partially supervised data. We then address optimizing these objectives end-to-end using backpropagation and the Relaxed Perturb-and-MAP algorithm. Finally, we propose a series of increasingly complex generative models to explore the potential of our modeling framework for improving tagging performance.

2.3.1 Semi-Supervised VAE

The purpose of this work is to consider methods for estimation of q_ϕ in semi-supervised data regimes, as in Kingma et al. (2014); Miao & Blunsom (2016); Yang et al. (2017), where there is additional unlabeled data $\mathcal{D}_U = \{(x^i_{1:N^i})\}$. To learn in this setting, we consider generative models of tags and tokens $p_\theta(x_{1:N}|y_{1:N})p(y_{1:N})$ and, for unobserved tags, aim to optimize the marginal likelihood of the observed tokens under the generative model.

$$\log p_\theta(x_{1:N}) = \log \sum_{y_{1:N}} p_\theta(x_{1:N}|y_{1:N})p(y_{1:N})$$

This marginalization is intractable for models that are not factored among y_i, so we resort to optimizing the familiar evidence lower bound (ELBO) (Jordan et al., 1999; Blei et al., 2017) with an approximate variational posterior distribution, which we set to our tagging model q_ϕ. We maximize the ELBO on unlabeled data in addition to maximum likelihood losses for both the inference and generative models on labeled data, yielding the following objectives:

$$\mathcal{L}_S = \sum_{(x,y)\in\mathcal{D}_S} \log p_\theta(x|y) + \log q_\phi(y|x) \tag{3}$$

$$\mathcal{L}_U = \sum_{x\in\mathcal{D}_U} \mathbb{E}_{q_\phi}[\log p_\theta(x|y)] - \beta \mathrm{KL}(q_\phi||p(y)) \tag{4}$$

$$\mathcal{L}(\theta, \phi; \mathcal{D}_S \cup \mathcal{D}_U, \alpha, \beta) = \mathcal{L}_S + \alpha\mathcal{L}_U \tag{5}$$

where α is scalar hyper-parameter used to balance the supervised loss $\mathcal{L}_S$ and the unsupervised loss $\mathcal{L}_U$ (Kingma et al., 2014). β is a scalar hyper-parameter used to balance the reconstruction and KL terms for the unsupervised loss (Bowman et al., 2015; Higgins et al., 2017). We note that, unlike a traditional VAE, this model contains no continuous latent variables.

[3] We omit the Pretrained-LM parameters since they are not updated during training.

2.3.2 PARTIALLY SUPERVISED LEARNING (PSL)

Assuming that supervised sentences are completely labeled is a restrictive setup for semi-supervised learning of a named entity tagger. It would be useful to be able to learn the tagger on sentences which are only *partially* labeled, where we only observe some named entity spans, but are not guaranteed all entity spans in the sentence are annotated and no O tags are manually annotated. [4] This presents a challenge in that we are no longer able to assume the usual implicit presence of O tags, since unannotated tokens are ambiguous. While it is possible to optimize the marginal likelihood of the CRF on only the observed tags $y_{\mathcal{O}}$, $\mathcal{O} \subset \{1, \ldots, N\}$ in the sentence (Tsuboi et al., 2008), doing so naively will result in a degenerate model that never predicts O, by far the most common tag (Jie et al., 2019). Interestingly, this scenario is easily addressed by the variational framework via the KL term. We do this by reformulating the objective in Equation 5 to account for partially observed tag sequences:

Let $\mathcal{D}_P = \{(x^i_{1:N_i}, y^i_{\mathcal{O}})\}$ be the partially observed dataset where, for some sentence i, $\mathcal{O} \subset \{1, \ldots, N_i\}$ is the set of observed positions and $\mathcal{U} = \{1, \ldots, N_i\} \setminus \mathcal{O}$ is the set of unobserved positions. Our partially supervised objective is then

$$\mathcal{L}_P = \sum_{(x, y_{\mathcal{O}}) \in \mathcal{D}_P} \Big[\log q_\phi(y_{\mathcal{O}}|x) + \alpha \mathbb{E}_{q_\phi}[\log p_\theta(x|y_{\mathcal{O}} \cup y_{\mathcal{U}})] - \alpha\beta \text{KL}(q_\phi(y_{\mathcal{U}}|x, y_{\mathcal{O}})||p(y_{\mathcal{U}})) \Big] \quad (6)$$

which can be optimized as before using the constrained forward-backward and KL algorithms detailed in Appendix B.

We also explore using this approach simply for regularization of the CRF posterior by omitting the token model $p_\theta(x|y)$. Since we do not have trainable parameters for the generative model in this case, the reconstruction likelihood drops out of the objective and we have, for a single datum $(x^i, y^i_{\mathcal{O}}) \in \mathcal{D}_P$, the following loss:

$$\mathcal{L}^i_P = \log q_\phi(y^i_{\mathcal{O}}|x^i) - \alpha\beta \text{KL}(q_\phi(y^i_{\mathcal{U}}|x^i, y^i_{\mathcal{O}})||p(y^i_{\mathcal{U}}))$$

2.3.3 DIFFERENTIABLE PERTURB-AND-MAP

Optimizing Equations 5 and 6 with respect to θ and ϕ using backpropagation and SGD is straightforward for every term except for the expectation terms $\mathbb{E}_{q_\phi(y|x)}[\log p_\theta(x|y)]$. To optimize these expectations, we first make an Monte Carlo approximation using a single sample drawn from q_ϕ. This discrete sample, however, is not differentiable with respect to ϕ and blocks gradient computation. While we may appeal to score function estimation (Miller, 1967; Williams, 1992; Paisley et al., 2012; Ranganath et al., 2014; Miao & Blunsom, 2016; Mohamed et al., 2019) to work around this, its high-variance gradients make successful optimization difficult.

Following Papandreou & Yuille (2011); Mensch & Blondel (2018); Corro & Titov (2018); Kim et al. (2019), we can compute approximate samples from q_ϕ that are differentiable with respect to ϕ using the Relaxed Perturb-and-MAP algorithm (Corro & Titov, 2018; Kim et al., 2019). Due to space limitations, we leave the derivation of Relaxed Perturb-and-MAP for linear-chain CRFs to Appendix A and detail the resulting CRF algorithms in Appendix B.

2.4 PROPOSED GENERATIVE MODELS

We model the prior distribution of tag sequences $y_{1:N}$ as the per-tag product of a fixed categorical distribution $p(y_{1:N}) = \prod_i p(y_i)$. The KL between q_ϕ and this distribution can be computed in polynomial time using a modification of the forward recursion derived in Mann & McCallum (2007), detailed in Appendix B.

We experiment with several variations of architectures for $p_\theta(x_{1:N}|y_{1:N})$, presented in order of increasing complexity.

Baseline - CRF-Autoencoder (AE): The CRF Autoencoder (Ammar et al., 2014; Zhang et al., 2017) is the previous state-of-the-art semi-supervised linear-chain CRF, which we consider a strong

[4] This regime applies to situations such as weak supervision (i.e. a low-recall database or gazatteer used for distant supervision), incidental supervision (i.e., a random Wikipedia sentence), or online and active learning.

baseline. This model uses a tractable, fully factored generative model of tokens given tags and does not require approximate inference. Due to space limitations, we have detailed our implementation in Appendix C.

MF: This is our simplest proposed generative model. We first embed the relaxed tag samples, represented as simplex vectors $y_i \in \Delta^{|\mathcal{Y}|}$, into $\mathbb{R}^{d_{y_p}}$ as the weighted combination of the input vector representations for each possible tag:

$$u_i = U y_i, \;\; U \in \mathbb{R}^{d_{y_p} \times |\mathcal{Y}|} \tag{7}$$

We then compute factored token probabilities with an inner product

$$p_\theta(x_i|y_i) = \sigma_{\mathcal{X}}(w_{x_i}^\top u_i)$$

where $\sigma_{\mathcal{X}}$ is the softmax function normalized over $\mathcal{X}$. This model is generalization of the CRF Autoencoder architecture in Appendix C where the tag-token parameters $\theta_{x,y}$ are computed with a low-rank factorization $WU^\top$.

MT: The restrictive factorization of **MF** is undesirable, since we expect that information about nearby tags may be discriminative of individual tokens. To test this, we extend **MF** to use the full tag context by encoding the embedded tag sequence jointly using a two-layer transformer (Vaswani et al., 2017) with four attention heads per layer before predicting the tokens independently. That is,

$$p_\theta(x_i|y_{1:N}) = \sigma_{\mathcal{X}}(w_{x_i}^\top v_i), \;\; v_{1:N} = \text{Transformer}_\theta(u_{1:N})$$

MF-GPT2: Next, we see if we can leverage information from a pretrained language model to provide additional training signal to p_θ. We extend **MF** by adding the fixed pretrained language modeling parameters from GPT2 to the token scores:

$$p_\theta(x_i|y_i, x_{<i}) = \sigma_{\mathcal{X}} \left(\frac{w_{x_i}^\top u_i}{\sqrt{d_{y_p}}} + \frac{z_{x_i}^\top h_i^0}{\sqrt{d_{\text{GPT2}}}} \right)$$

where z_{x_i} and h_i^0 are the input token embeddings and hidden states from GPT2, respectively. We additionally normalize the scales of the factors by the square root of the vector dimensionalities to prevent the GPT2 scores from washing out the tag-encoding scores ($d_{y_p} = 300$ and $d_{GPT2} = 768$).

MT-GPT2: We add the same autoregressive extention to **MT**, using the tag encodings v instead of embeddings u.

$$p_\theta(x_i|y_{1:N}, x_{<i}) = \sigma_{\mathcal{X}} \left(\frac{w_{x_i}^\top v_i}{\sqrt{d_{y_p}}} + \frac{z_{x_i}^\top h_i^0}{\sqrt{d_{\text{GPT2}}}} \right)$$

MT-GPT2-PoE: We also consider an autoregressive extension of **MT**, similar to **MT-GPT2**, that uses a product of experts (PoE) (Hinton, 2002) factorization instead

$$p_\theta(x_i|y, x_{<i}) = \sigma_{\mathcal{X}} \big(p_\theta(x_i|y_{1:N}) p_{\text{GPT2}}(x_i|x_{<i}) \big)$$

$$p_\theta(x_i|y_{1:N}) = \sigma_{\mathcal{X}}(w_{x_i}^\top v_i), \;\; p_{\text{GPT2}}(x_i|x_{<i}) = \sigma_{\mathcal{X}}(z_{x_i}^\top h_i^0)$$

MT-GPT2-Residual: Our last variation directly couples GPT2 with p_θ by predicting a residual via a two-layer MLP based on the tag encoding and GPT2 state:

$$p_\theta(x_i|y_{1:N}, x_{<i}) = \sigma_{\mathcal{X}}(z_{x_i}^\top \bar{h}_i^0), \;\; \bar{h}_i^0 = h_i^0 + f_{\text{MLP}}(\langle h_i^0, v_i \rangle)$$

For the **MF-GPT2**, **MT-GPT2**, and **MT-GPT2-PoE** models, we choose these factorizations specifically to prevent the trainable parameters from conditioning on previous word information, removing the possibility of the model learning to ignore the noisy latent tags in favor of the strong signal provided by pretrained encodings of the sentence histories (Bowman et al., 2015; Yang et al., 2017; Kim et al., 2018). We further freeze the GPT2 parameters for all models, forcing the only path for improving the generative likelihood to be through the improved estimation and encoding of the tags $y_{1:N}$.

3 EXPERIMENTS AND RESULTS

We experiment first with the proposed models generative models for SSL and PSL in a moderately resourced regime (keeping 10% labeled data) to explore their relative merits. We then evaluate our best generative model from these experiments, (**MT**), with an improved bidirectional encoder language model in a low- and high-resource settings, varying the amount of unlabeled data.[5]

For data, we use the OntoNotes 5 (Hovy et al., 2006) NER corpus, which consists of 18 entity types annotated in 82,120 train, 12,678 validation, and 8,968 test sentences.

3.1 EXPLORATION OF GENERATIVE ARCHITECTURES

We begin by comparing the proposed generative models, **M*** along with the following baselines:

1. **Supervised (S)**: The supervised tagger trained only on the 10% labeled data.
2. **Supervised 100% (S*)**: The supervised tagger trained on the 100% labeled data, used for quantifying the performance loss from using less data.
3. **AE-Exact**: The CRF Autoencoder using exact inference (detailed in Appendix C.)
4. **AE-Approx**: The same tag-token pair parameterization used by the CRF Autoencoder, but trained with the approximate ELBO objective as in Equation 11 instead of the exact objective in Equation 12. The purpose here is to see if we lose anything by resorting to the approximate ELBO objective.

To simulate moderate-resource SSL, we keep annotations for only 10% of the sentences, yielding $8,212$ labeled sentences with $13,025$ annotated spans and $73,908$ unlabeled sentences. Results are shown in Table 1. All models except **S*** use this 10% labeled data.

We first evaluate the proposed models and baselines without the use of a prior, since the use of a locally normalized factored prior can encourage overly uncertain joint distributions and degrade performance (Jiao et al., 2006; Mann & McCallum, 2007; Corro & Titov, 2018). We then explore the inclusion of the priors for the supervised and **MT** models with $\beta = 0.01$.[6]

We explore two varieties of prior tag distributions: (1) the "gold" empirical tag distribution (Emp) from the full training dataset and (2) a simple, but informative, hand-crafted prior (Sim) that places 50% mass on the O tag and distributes the rest of its mass evenly among the remaining tags. We view (2) as a practical approach, since it does not require knowledge of the gold tag distribution, and use (1) to quantify any relative disadvantage from not using the gold prior. We find that including the prior with a small weight, $\beta = 0.01$, marginally improved performance and interestingly, the simple prior outperforms the empirical prior, most likely because it is slightly smoother and does not emphasize the O tag as heavily.[7]

Curiously, we found that the approximate training of the CRF Autoencoder **AE-Approx** outperformed the exact approach **AE-Exact** by nearly 2% F1.

We also note that our attempts to leverage signal from the pretrained autoregressive GPT2 states had negligible or negative effects on performance, thus we conclude that it is the addition of the joint encoding transformer architecture **MT** that provides the most gains ($+0.8\%$ F1).

PSL: We also evaluate the supervised and transformer-based generative models, **S** and **MT**, on the more difficult PSL setup, where naively training the supervised model on the marginal likelihood of observed tags produces a degenerate model, due to the observation bias of never having O tags. In this setting we drop 90% of the annotations from sentences randomly, resulting in 82,120 incompletely annotated sentences with 12,883 annotations total. We compare the gold and simple priors for each model. From the bottom of Table 1, we see that again our proposed transformer model **MT** outperforms the supervised-only model, this time by $+1.3\%$ F1. We also find that in this case, the **MT** models need to be trained with higher prior weights $\beta = 0.1$, otherwise they diverge towards using the O tag more uniformly with the other tags to achieve better generative likelihoods.

[5]Code and experiments are available online at `github.com/<anonymizedforsubmission>`

[6]In preliminary SSL experiments we found $\beta > 0.01$ to have a negative impact on performance, likely due to global/local normalization mismatch of the CRF and the prior.

[7]The empirical prior puts 85% mass on the O tag

Model	α	β	P	R	F1
Supervised 100% (S*)	-	0.0	0.808	0.798	0.803
Supervised (S)	-	0.0	0.761	0.738	0.749
AE-Exact	-	0.0	0.736	0.721	0.728
AE-Approx	0.1	0.0	**0.767**	0.728	0.747
MF (Factored)	0.1	0.0	0.761	0.719	0.739
MT (Transformer)	0.1	0.0	0.758	0.756	0.757
MF-GPT2	0.1	0.0	0.754	0.710	0.731
MT-GPT2	0.1	0.0	0.762	0.755	0.759
MT-GPT2 (no-scale)	0.1	0.0	0.766	0.734	0.751
MT-GPT2-PoE	0.1	0.0	0.766	0.742	0.753
MT-GPT2-Residual	0.1	0.0	0.766	0.740	0.753
S (Emp)	0.1	0.01	0.754	0.733	0.743
S (Sim)	0.1	0.01	0.754	0.734	0.743
MT (Emp)	0.1	0.01	0.760	0.756	0.758
MT (Sim)	0.1	0.01	0.762	**0.757**	**0.760**
S (Emp) - PSL	0.1	0.01	**0.741**	0.725	0.733
S (Sim) - PSL	0.1	0.01	0.730	0.740	0.735
MT (Emp) - PSL	0.1	0.1	0.731	0.761	0.746
MT (Sim) - PSL	0.1	0.1	0.724	**0.774**	**0.748**

Table 1: Semi-supervised and partially-supervised models on 10% supervised training data: best in bold, second best underlined. The proposed **MT*** improves performance in SSL and PSL by $+1.1\%$ F1 and $+1.3\%$ F1, respectively.

3.2 Varying Resources

Next we explore our best proposed architecture **MT** and the supervised baseline in low- and high-resource settings (1% and 100% training data, respectively) and study the effects of training with an additional 100K unlabeled sentences sampled from Wikipedia (detailed in Appendix E).

Since we found no advantage from using pretrained GPT2 information in the previous experiment, we evaluate the use of the bidirectional pretrained language model, RoBERTa (Liu et al., 2019b), since we expect bidirectional information to highly benefit performance (Strubell et al. (2017); Akbik et al. (2018), among others). We also experiment with a higher-capacity tagging model, **S-LG**, by adding more trainable Transformers ($L = 4$, $A = 8$, $H = 1024$) between the RoBERTa encodings and down-projection layers.

From Table 2 we see that, like in the 10% labeled data setting, the CRF-VAE improves upon the supervised model by 0.9% F1 in this 1% setting, but we find that including additional data from Wikipedia has a negative impact. A likely reason for this is the domain mismatch between Ontonotes5 and Wikipedia (news and encyclopedia, respectively).

In the high-resource setting, we find that using RoBERTa significantly improves upon GPT2 ($+5.7\%$ F1) and the additional capacity of **S-LG** further improves performance by $+2.2\%$ F1. Although we do not see a significant improvement from semi-supervised training with Wikipedia sentences, our model is competitive with previous state-of-the-art NER approaches and outperforms all previous approaches that do not use additional labeled data or gazetteers.

4 Related Work

Utilizing unlabeled data for semi-supervised learning in NER has been studied considerably in the literature. A common approach is a two-stage process where useful features are learned from unsupervised data, then incorporated into models which are then trained only on the supervised data (Fernandes & Brefeld, 2011; Kim et al., 2015). With the rise of neural approaches, large-scale word vector (Mikolov et al., 2013; Pennington et al., 2014) and language model pretraining methods (Peters et al., 2018; Akbik et al., 2018; Devlin et al., 2018) can be regarded in the same vein.

Model	$\lvert\mathcal{D}_S\rvert$	$\lvert\mathcal{D}_U\rvert$	α	β	P	R	F1
S	13K	0	-	-	0.744	0.712	0.728
MT (Sim)	13K	82K	0.01	0.1	0.752	0.739	0.737
MT (Sim) + Wiki	13K	182K	0.01	0.1	0.746	0.721	0.733
Strubell et al. (2017)	82K	0	-	-	-	-	0.869
Clark et al. (2018)†	82K	>1M	-	-	-	-	0.888
Chen et al. (2019)	82K	0	-	-	0.878	0.876	0.877
Akbik et al. (2018)†	95K	0	-	-	-	-	0.891
Liu et al. (2019a)†	82K	0	-	-	-	-	0.899
S (GPT2)	82K	0	-	-	0.808	0.798	0.803
S	82K	0	-	-	0.864	0.855	0.860
S-LG	82K	0	-	-	0.873	**0.892**	0.882
MT-LG (Sim) + Wiki	82K	182K	0.1	0.01	**0.880**	0.890	**0.884**

Table 2: Low- and high-resource results with RoBERTa, varying available unlabeled data. Best scores not using additional labeled data in bold. † Uses additional labeled data or gazetteers.

Another approach is to automatically create silver-labeled data using outside resources, whose low recall induces a partially supervised learning problem. Bellare & McCallum (2007) approach the problem by distantly supervising (Mintz et al., 2009) spans using a database. Carlson et al. (2009) similarly use a gazetteer and adapt the structured perceptron (Collins, 2002) to handle partially labeled sequences, while Yang et al. (2018) optimize the marginal likelihood (Tsuboi et al., 2008) of the distantly annotated tags. Yang et al. (2018)'s method, however, still requires some fully labeled data to handle proper prediction of the O tag. The problem setup from Jie et al. (2019) is the same as our PSL regime, but they use a cross-validated self-training approach. Greenberg et al. (2018) use a marginal likelihood objective to pool overlapping NER tasks and datasets, but must exploit dataset-specific constraints, limiting the allowable latent tags to debias the model from never predicting O tags.

Generative latent-variable approaches also provide an attractive approach to learning on unsupervised data. Ammar et al. (2014) present an approach that uses the CRF for autoencoding and Zhang et al. (2017) extend it to neural CRFs, but both require the use of a restricted factored generative model to make learning tractable. Deep generative models of text have shown promise in recent years, with demonstrated applications to document representation learning (Miao et al., 2016), sentence generation (Bowman et al., 2015; Yang et al., 2017; Kim et al., 2018), compression (Miao & Blunsom, 2016), translation (Deng et al., 2018), and parsing (Corro & Titov, 2018). However, to the best of our knowledge, this framework has yet to be utilized for NER and tagging CRFs. A key challenge for learning VAEs with discrete latent variables is optimization with respect to the inference model parameters ϕ. While we may appeal to score function estimation (Williams, 1992; Paisley et al., 2012; Ranganath et al., 2014; Miao & Blunsom, 2016), its empirical high-variance gradients make successful optimization difficult. Alternatively, obtaining gradients with respect to ϕ can be achieved using the relaxed Gumbel-max trick (Jang et al., 2016; Maddison et al., 2016) and has been recently extended to latent tree-CRFs by (Corro & Titov, 2018), which we make use of here for sequence CRFs.

5 Conclusions

We proposed a novel generative model for semi-supervised learning in NER. By treating a neural CRF as the amortized variational posterior in the generative model and taking relaxed differentiable samples, we were able to utilize a transformer architecture in the generative model to condition on more context and provide appreciable performance gains over supervised and strong baselines on both semi-supervised and partially-supervised datasets. We also found that inclusion of powerful pretrained autoregressive language modeling states had neglible or negative effects while using a pretrained bidirectional encoder offers significant performance gains. Future work includes the use of larger in-domain unlabeled corpora and the inclusion of latent-variable CRFs in more interesting joint semi-supervised models of annotations, such as relation extraction and entity linking.

REFERENCES

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1638–1649, 2018.

Waleed Ammar, Chris Dyer, and Noah A Smith. Conditional random field autoencoders for unsupervised structured prediction. In *Advances in Neural Information Processing Systems*, pp. 3311–3319, 2014.

Kedar Bellare and Andrew McCallum. Learning extractors from unlabeled text using relevant databases. In *Sixth international workshop on information integration on the web*, 2007.

James O Berger. *Statistical decision theory and Bayesian analysis; 2nd ed.* Springer Series in Statistics. Springer, New York, 1985. doi: 10.1007/978-1-4757-4286-2. URL `https://cds.cern.ch/record/1327974`.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Andrew Carlson, Scott Gaffney, and Flavian Vasile. Learning a named entity tagger from gazetteers with the partial perceptron. In *AAAI Spring Symposium: Learning by Reading and Learning to Read*, pp. 7–13, 2009.

Hui Chen, Zijia Lin, Guiguang Ding, Jianguang Lou, Yusen Zhang, and Borje Karlsson. Grn: Gated relation network to enhance convolutional neural network for named entity recognition. In *Proceedings of AAAI*, 2019.

Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*, 2018.

Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pp. 1–8. Association for Computational Linguistics, 2002.

Caio Corro and Ivan Titov. Differentiable perturb-and-parse: Semi-supervised parsing with a structured variational autoencoder. *arXiv preprint arXiv:1807.09875*, 2018.

Yuntian Deng, Yoon Kim, Justin Chiu, Demi Guo, and Alexander Rush. Latent alignment and variational attention. In *Advances in Neural Information Processing Systems*, pp. 9712–9724, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Eraldo R Fernandes and Ulf Brefeld. Learning from partially annotated sequences. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 407–422. Springer, 2011.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. Marginal likelihood training of bilstm-crf for biomedical named entity recognition from disjoint label sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2824–2829, 2018.

Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, volume 3, 2017.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pp. 57–60, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1614049.1614064.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

Inc. Hugging Face. PyTorch Pretrained BERT: The Big & Extending Repository of pretrained Transformers, May 2019. URL https://github.com/huggingface/pytorch-pretrained-BERT.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Feng Jiao, Shaojun Wang, Chi-Hoon Lee, Russell Greiner, and Dale Schuurmans. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 209–216. Association for Computational Linguistics, 2006.

Zhanming Jie, Pengjun Xie, Wei Lu, Ruixue Ding, and Linlin Li. Better modeling of incomplete annotations for named entity recognition. In *Proceedings of NAACL*, 2019.

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

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M Rush. Structured attention networks. *arXiv preprint arXiv:1702.00887*, 2017.

Yoon Kim, Sam Wiseman, Andrew C Miller, David Sontag, and Alexander M Rush. Semi-amortized variational autoencoders. *arXiv preprint arXiv:1802.02550*, 2018.

Yoon Kim, Alexander M Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised recurrent neural network grammars. *arXiv preprint arXiv:1904.03746*, 2019.

Young-Bum Kim, Minwoo Jeong, Karl Stratos, and Ruhi Sarikaya. Weakly supervised slot tagging with partially labeled sequences from web search click logs. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 84–92, 2015.

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

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

Tianyu Liu, Jin-ge Yao, and Chin-Yew Lin. Towards improving neural named entity recognition with gazetteers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5301–5307, 2019a.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019b.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Gideon S Mann and Andrew McCallum. Efficient computation of entropy gradient for semi-supervised conditional random fields. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pp. 109–112. Association for Computational Linguistics, 2007.

Arthur Mensch and Mathieu Blondel. Differentiable dynamic programming for structured prediction and attention. *arXiv preprint arXiv:1802.03676*, 2018.

Yishu Miao and Phil Blunsom. Language as a latent variable: Discrete generative models for sentence compression. *arXiv preprint arXiv:1609.07317*, 2016.

Yishu Miao, Lei Yu, and Phil Blunsom. Neural variational inference for text processing. In *International conference on machine learning*, pp. 1727–1736, 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

Laurence B Miller. Monte carlo analysis of reactivity coefficients in fast reactors general theory and applications. Technical report, Argonne National Lab.(ANL), Argonne, IL (United States), 1967.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pp. 1003–1011. Association for Computational Linguistics, 2009.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *arXiv preprint arXiv:1906.10652*, 2019.

John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *arXiv preprint arXiv:1206.6430*, 2012.

George Papandreou and Alan L Yuille. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models. In *2011 International Conference on Computer Vision*, pp. 193–200. IEEE, 2011.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110(1):43–70, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1:8, 2019.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the thirteenth conference on computational natural language learning*, pp. 147–155. Association for Computational Linguistics, 2009.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pp. 254–263. Association for Computational Linguistics, 2008.

Emma Strubell, Patrick Verga, David Belanger, and Andrew McCallum. Fast and accurate entity recognition with iterated dilated convolutions. *arXiv preprint arXiv:1702.02098*, 2017.

Yuta Tsuboi, Hisashi Kashima, Hiroki Oda, Shinsuke Mori, and Yuji Matsumoto. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pp. 897–904. Association for Computational Linguistics, 2008.

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

Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. Distantly supervised ner with partial annotation learning and reinforcement learning. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2159–2169, 2018.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3881–3890. JMLR. org, 2017.

Xiao Zhang, Yong Jiang, Hao Peng, Kewei Tu, and Dan Goldwasser. Semi-supervised structured prediction with neural crf autoencoder. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1701–1711, 2017.

A RELAXED PERTURB-AND-MAP FOR LINEAR CHAIN CRFS

Let $\tilde{q}_\phi(y|x;\tau)$ be the distribution on y with the potentials ψ for each tag at each position perturbed by Gumbel noise $\gamma \overset{\text{iid}}{\sim} \mathcal{G}(0,1)$ (Gumbel, 1954) and $\tau \geq 0$ be the temperature:

$$\tilde{q}_\phi(y|x;\tau) = \frac{\exp\left\{(\sum_{i=0}^{N} \psi_{y_i,y_{i+1}} + \gamma_{y_i})/\tau\right\}}{\sum_{y'_{1:N}} \exp\left\{(\sum_{i=0}^{N} \psi_{y'_i,y'_{i+1}} + \gamma_{y'_i})/\tau)\right\}}$$

We know from Papandreou & Yuille (2011) that the MAP sequence from this perturbed distribution is a *sample* from the unperturbed distribution. Coupled with the property that the zero temperature limit of the Gibbs distribution is the MAP state (Wainwright et al., 2008), it immediately follows that the zero temperature limit of the perturbed $\tilde{q}$ is a sample from q:

$$\tilde{y} = \arg\max_{y \in \mathcal{Y}} \tilde{q}_\phi(y|x;\tau) \tag{8}$$

$$\lim_{\tau \to 0} q_\phi(y|x;\tau) = \text{one-hot}(\arg\max_{y \in \mathcal{Y}} q_\phi(y|x)) \tag{9}$$

$$\Rightarrow \lim_{\tau \to 0} \tilde{q}_\phi(y|x;\tau) = \text{one-hot}(\tilde{y}) \tag{10}$$

where $q_\phi(y|x;\tau)$ is the tempered but unperturbed q_ϕ and "one-hot" is a function that converts elements of $\mathcal{Y}^N$ to a one-hot vector representation.

Thus we can use the temperature τ to anneal the perturbed joint distribution $\tilde{q}_\phi(y|x;\tau)$ to a sample from the unperturbed distribution, $\tilde{y} \sim q_\phi$. When $\tau > 0$, $\tilde{q}_\phi(y|x;\tau)$ is differentiable and can be used for end-to-end optimization by allowing us to approximate the expectation with a relaxed single-sample Monte Carlo estimate:

$$\mathbb{E}_{q_\phi(y|x)}[\log p_\theta(x|y)] \approx \log p_\theta(x|\tilde{q}_\phi(y|x;\tau)) \tag{11}$$

where we have modified $\log p_\theta(x|y)$ to accept the simplex representations of $y_{1:N}$ from $\tilde{q}_\phi$ instead of discrete elements, which has the effect of $\log p_\theta(x|y)$ computing a weighted combination of its input vector representations for $y \in \mathcal{Y}$ similarly to an attention mechanism or the annotation function in Kim et al. (2017) (see Equation 7.)

This can be thought of as a generalization of the Gumbel-softmax trick from Jang et al. (2016); Maddison et al. (2016) to structured joint distributions.

The statements in (8-10) also imply something of practical interest: we can compute (1) the argmax (Viterbi decoding) and its differentiable relaxation; (2) a sample and its differentiable relaxation; (3) the partition function; and (4) the marginal tag distributions, all using the same sum-product algorithm implementation, controlled by the temperature and the presence of noise. We have detailed the algorithm in Appendix B.

B CRF ALGORITHMS

In Algorithm 1 we have detailed the stable, log-space implementation of the generalized forward-backward algorithm for computing (1) the argmax (Viterbi decoding) and its differentiable relaxation; (2) a sample and its differentiable relaxation; (3) the partition function; and (4) the marginal tag distributions below. While this algorithm does provide practical convenience, we note that real implementations should have separate routines for computing the partition function (running only the forward algorithm), and the discrete $\tau = 0$ Viterbi algorithm, since it is more numerically stable and efficient.

We also have included the dynamic program for computing the constrained KL divergence between q_ϕ and a factored $p(y)$ in Algorithm 2.

C CRF AUTOENCODER

The idea of using a CRF to reconstruct tokens given tags for SSL has been explored before by Ammar et al. (2014); Zhang et al. (2017), which we consider to be a strong baseline and restate

Algorithm 1 Relaxed, Constrained, Perturbed Forward-Backward

Notation: $\underset{x}{\text{LSE}} := \log \sum_x \exp$

Input: Local potentials $\Psi_{y_{1:N}}$, transition potentials $\Psi_{y,y'}$, *perturb* boolean, temperature τ, the special start symbol and end symbols $*, \diamond$, and the set of allowable tags for each position $\mathcal{Y}_i \subseteq \mathcal{Y}$ (allows for partially observed/constrained sequences.)

Procedure:

$\log \alpha[0, y] \leftarrow \psi_{*,y}/\tau, \quad \log \beta[N+1, y] \leftarrow \psi_{y,\diamond}/\tau$ ▷ Initialize recursions bases

if *perturb* **then**

  $\psi_{y_i} \leftarrow \psi_{y_i} + \gamma_{y_i}, \;\; \gamma_{y_i} \overset{\text{iid}}{\sim} \mathcal{G}(0,1)$ ▷ Perturb local potentials

end if

for $i = 1, \ldots, N$ **do** ▷ Compute forward lattice

  for $y \in \mathcal{Y}_i$ **do**

    $\log \alpha[i, y] \leftarrow \underset{y' \in \mathcal{Y}_{i-1}}{\text{LSE}} \; (\psi_{y',y} + \psi_y)/\tau + \log \alpha[i-1, y']$

  end for

end for

for $i = N, \ldots, 1$ **do** ▷ Compute backward lattice

  for $y \in \mathcal{Y}_i$ **do**

    $\log \beta[i, y] \leftarrow \underset{y' \in \mathcal{Y}_{i+1}}{\text{LSE}} \; (\psi_{y,y'} + \psi_y)/\tau + \log \beta[i+1, y']$

  end for

end for

$\mu_{y_i} \leftarrow \sigma_{\mathcal{Y}_i}\big(\underset{y_{i+1} \in \mathcal{Y}_{i+1}}{\text{LSE}} \log \alpha[i, y_i] + (\psi_{y_i} + \psi_{y_i,y_{i+1}})/\tau + \log \beta[i+1, y_{i+1}]\big)$ ▷ Tag marginals

Output:

If *perturb* **then**

  Relaxed sample $\tilde{q}_\phi(y|x;\tau) \leftarrow \mu_{y_{1:N}}$ ▷ Converges to sample at $\tau = 0$

Else if $\tau = 1$ **then**

  Partition function $Z(\psi) \leftarrow \sum_{y' \in \mathcal{Y}_N} \exp\{\psi_{y',\diamond} + \log \alpha[N, y']\}$

  Tag marginals $q_\phi(y|x) \leftarrow \mu_{y_{1:N}}$

Else

  Relaxed argmax $q_\phi(y|x;\tau) \leftarrow \mu_{y_{1:N}}$ ▷ Converges to Viterbi at $\tau = 0$

Algorithm 2 Constrained KL

Notation: $\underset{x}{\text{LSE}} := \log \sum_x \exp$

Input: Local potentials $\Psi_{y_{1:N}}$, transition potentials $\Psi_{y,y'}$, prior distribution $p(y_{1:N}) = \prod p(y_i)$, the special start symbol and end symbols $*, \diamond$, and the set of allowable tags for each position $\mathcal{Y}_i \subseteq \mathcal{Y}$ (allows for partially observed/constrained sequences.)

Procedure:

$\log \alpha[0, y] \leftarrow \psi_{*,y}, \quad \text{KL}^\alpha[0, y] \leftarrow 0 \;\; \forall y \in \mathcal{Y}_1$ ▷ Initialize recursions bases

for $i = 1, \ldots, N$ **do**

  for $y_i \in \mathcal{Y}_i$ **do** ▷ Same as forward algorithm

    $\log \alpha[i, y_i] \leftarrow \underset{y_{i-1} \in \mathcal{Y}_{i-1}}{\text{LSE}} \; (\psi_{y_{i-1},y_i} + \psi_{y_i}) + \log \alpha[i-1, y_{i-1}]$

  end for

  for $y_{i+1} \in \mathcal{Y}_{i+1}$ **do** ▷ Compute KL lattice

    $q(y_i|y_{i+1}) \leftarrow \sigma_{\mathcal{Y}_i}(\log \alpha[i, y_i] + \psi_{y_i,y_{i+1}} + \psi_{y_{i+1}})$

    $\text{KL}^\alpha[i, y_{i+1}] \leftarrow \sum_{y_i \in \mathcal{Y}_i} q(y_i|y_{i+1})\big[\log q(y_i|y_{i+1}) - \log p(y_i) + \text{KL}^\alpha[i-1, y_i]\big]$

  end for

end for

Output: $\text{KL}(q||p) \leftarrow \text{KL}^\alpha[N, \diamond]$

here for clarity. Termed the CRF Autoencoder, the model treats the the tags as intermediate latent variables in a conditional model and optimizes the marginal conditional likelihood of reconstructing the input.

$$\log p(\hat{x}|x) = \log \sum_{y_{1:N}} p_\theta(\hat{x}|y_{1:N}) q_\phi(y_{1:N}|x)$$

By judiciously choosing $p_\theta(\hat{x}|y)$ to be factored among positions i, we can compute the marginal reconstruction likelihood exactly:

$$\begin{aligned}
\log & p(\hat{x}|x) = \log \sum_{y_{1:N}} q_\phi(y_{1:N}|x) \prod_{i=1}^{N} p_\theta(\hat{x}_i|y_i) \\
&= \log \sum_{y_{1:N}} \exp\{\sum_{i=0}^{N} \psi_{y_i, y_{i+1}} + \log p_\theta(\hat{x}_i|y_i)\} - \log Z(\psi) \\
&= \log Z(\psi + \log p_\theta) - \log Z(\psi) \qquad (12)
\end{aligned}$$

where $\log Z(\psi + \log p_\theta)$ is a slight abuse of notation intended to illustrate that the first term in Equation 12 is the same computation as the partition function, but with the generative log-likelihoods added to the CRF potentials.

We note that instead of using the Mixed-EM procedure from Zhang et al. (2017), we model $p_\theta(\hat{x}_i|y_i)$ using free logit parameters $\theta_{x,y}$ for each token-tag pair and normalize using a softmax, which allows for end-to-end optimization via backpropagation and SGD.

D Experiment Hyperparameter and Optimization Settings

We train each model to convergence using early-stopping on the F1 score of the validation data, with a patience of 10 epochs. For all models that do not have trainable transformers, we train using the Adam optimizer (Kingma & Ba, 2014) with a learning rate of 0.001, and a batch size of 128. For those with transformers (**MT***), we train using Adam, a batch size of 32, and the Noam learning rate schedule from Vaswani et al. (2017) with a model size of $d_{y_p} = 300$ and $16,000$ warm-up steps (Popel & Bojar, 2018).

Additionally, we use gradient clipping of 5 for all models and a temperature of $\tau = .66$ for all relaxed sampling models. We implemented our models in PyTorch (Paszke et al., 2017) using the AllenNLP framework (Gardner et al., 2018) and the Hugging Face (2019) implementation of the pretrained GPT2 and RoBERTa.

We have made all code, data, and experiments available online at `github.com/<anonymizedforsubmission>` for reproducibility and reuse. All experimental settings can be reproduced using the configuration files in the repo.

E Gathering Additional Unlabeled Data

For the experiments in §3.2, we gather an additional training corpus of out-of-domain encyclopedic sentences from Wikipedia. To try to get a sample that better aligns with the Ontonotes5 data, these sentences were gathered with an informed process, which was performed as follows:

1. Using the repository *<anonymized for submission>*, we extract English Wikipedia and align it with Wikidata.
2. We then look up the entity classes from the Ontonotes5 specification (Hovy et al., 2006) in Wikidata and, for each NER class, find all Wikidata classes that are below this class in ontology (all subclasses).
3. We then find all items which are instances of these classes and also have Wikipedia pages. These are the Wikipedia entities which are likely to be instances of the NER classes.

4. Finally, we scan Wikipedia, mapping any available links to these NER classes, and keep the top 100K sentences according to the number of found annotations/token – the most ”densely” annotated sentences.