Variational Sequential Labelers for Semi-Supervised Learning

Mingda Chen **Qingming Tang** **Karen Livescu** **Kevin Gimpel**
Toyota Technological Institute at Chicago, Chicago, IL, 60637, USA
{mchen,qmtang,klivescu,kgimpel}@ttic.edu

Abstract

We introduce a family of multitask variational methods for semi-supervised sequence labeling. Our model family consists of a latent-variable generative model and a discriminative labeler. The generative models use latent variables to define the conditional probability of a word given its context, drawing inspiration from word prediction objectives commonly used in learning word embeddings. The labeler helps inject discriminative information into the latent space. We explore several latent variable configurations, including ones with hierarchical structure, which enables the model to account for both label-specific and word-specific information. Our models consistently outperform standard sequential baselines on 8 sequence labeling datasets, and improve further with unlabeled data.

1 Introduction

Sequence labeling tasks in natural language processing (NLP) often have limited annotated data available for model training. In such cases regularization can be important, and it can be helpful to use additional unlabeled data. One approach for both regularization and semi-supervised training is to design latent-variable generative models and then develop neural variational methods for learning and inference (Kingma and Welling, 2014; Rezende and Mohamed, 2015).

Neural variational methods have been quite successful for both generative modeling and representation learning, and have recently been applied to a variety of NLP tasks (Mnih and Gregor, 2014; Bowman et al., 2016; Miao et al., 2016; Serban et al., 2017; Zhou and Neubig, 2017; Hu et al., 2017). They are also very popular for semi-supervised training; when used in such scenarios, they typically have an additional task-specific prediction loss (Kingma et al., 2014; Maale et al., 2016; Zhou and Neubig, 2017; Yang et al., 2017b). However, it is still unclear how to use such methods in the context of sequence labeling.

In this paper, we apply neural variational methods to sequence labeling by combining a latent-variable generative model and a discriminatively-trained labeler. We refer to this family of procedures as variational sequential labelers (VSLs). Learning maximizes the conditional probability of each word given its context and minimizes the classification loss given the latent space. We explore several models within this family that use different kinds of conditional independence structure among the latent variables within each time step. Intuitively, the multiple latent variables can disentangle information pertaining to label-oriented and word-specific properties.

We study VSLs in the context of named entity recognition (NER) and several part-of-speech (POS) tagging tasks, both on English Twitter data and on data from six additional languages. Without unlabeled data, our models consistently show 0.5-0.8% accuracy improvements across tagging datasets and 0.8 F_1 improvement for NER. Adding unlabeled data further improves the model performance by 0.1-0.3% accuracy or 0.2 F_1 score. We obtain the best results with a hierarchical structure using two latent variables at each time step.

Our models, like generative latent variable models in general, have the ability to naturally combine labeled and unlabeled data. We obtain small but consistent performance improvements by adding unlabeled data. In the absence of unlabeled data, the variational loss acts as regularizer on the learned representation of the supervised sequence prediction model. Our results demonstrate that this regularization improves performance even when only labeled data is used. We also compare different ways of applying the classification loss when using a latent variable hierar-

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 215–226
Brussels, Belgium, October 31 - November 4, 2018.

chy, and find that the most effective structure also provides the cleanest separation of information in the latent space.

2 Related Work

There is a growing amount of work applying neural variational methods to NLP tasks, including document modeling (Mnih and Gregor, 2014; Miao et al., 2016; Serban et al., 2017), machine translation (Zhang et al., 2016), text generation (Bowman et al., 2016; Serban et al., 2017; Hu et al., 2017), language modeling (Bowman et al., 2016; Yang et al., 2017b), and sequence transduction (Zhou and Neubig, 2017), but we are not aware of any such work for sequence labeling. Before the advent of neural variational methods, there were several efforts in latent variable modeling for sequence labeling (Quattoni et al., 2007; Sun and Tsujii, 2009).

There has been a great deal of work on using variational autoencoders in semi-supervised settings (Kingma et al., 2014; Maale et al., 2016; Zhou and Neubig, 2017; Yang et al., 2017b). Semi-supervised sequence labeling has a rich history (Altun et al., 2006; Jiao et al., 2006; Mann and McCallum, 2008; Subramanya et al., 2010; Søgaard, 2011). The simplest methods, which are also popular currently, use representations learned from large amounts of unlabeled data (Miller et al., 2004; Owoputi et al., 2013; Peters et al., 2017). Recently, Zhang et al. (2017) proposed a structured neural autoencoder that can be jointly trained on both labeled and unlabeled data.

Our work involves multi-task losses and is therefore also related to the rich literature on multi-task learning for sequence labeling (Plank et al., 2016; Augenstein and Søgaard, 2017; Bingel and Søgaard, 2017; Rei, 2017, *inter alia*).

Another related thread of work is learning interpretable latent representations. Zhou and Neubig (2017) factorize an inflected word into lemma and morphology labels, using continuous and categorical latent variables. Hu et al. (2017) interpret a sentence as a combination of an unstructured latent code and a structured latent code, which can represent attributes of the sentence.

There have been several efforts in combining variational autoencoders and recurrent networks (Gregor et al., 2015; Chung et al., 2015; Fraccaro et al., 2016). While the details vary, these models typically contain latent variables at each time step in a sequence. This prior work mainly focused on ways of parameterizing the time dependence between the latent variables, which gives them more power in modeling distributions over observation sequences. In this paper, we similarly use latent variables at each time step, but we adopt stronger independence assumptions which leads to simpler models and inference procedures. Also, the models cited above were developed for modeling data distributions, rather than for supervised or semi-supervised learning, which is our focus here.

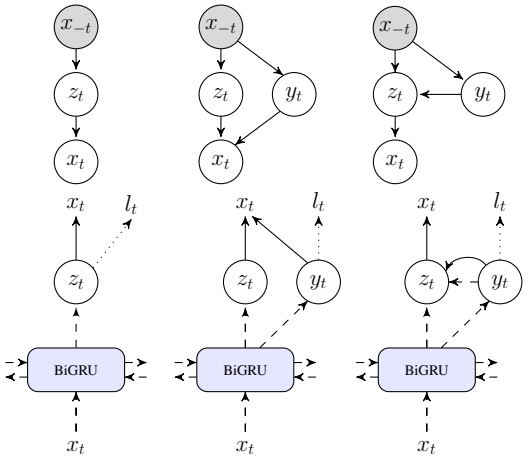
The key novelties in our work compared to the prior work mentioned above are the proposed sequential variational labelers and the investigation of latent variable hierarchies within these models. The empirical effectiveness of latent hierarchical structure in variational modeling is a key contribution of this paper and may be applicable to the other applications discussed above. Recent work, contemporaneous with this submission, similarly showed the advantages of combining hierarchical latent variables and variational learning for conversational modeling, in the context of a non-sequential model (Park et al., 2018).

3 Proposed Methods

We begin by describing variational autoencoders and the notation we will use in the following sections. We denote the input word sequence by $x_{1:T}$, the corresponding label sequence by $l_{1:T}$, the input words other than the word at position t by x_{-t}, the generative model by $p_\theta(\cdot)$, and the posterior inference model by $q_\phi(\cdot)$.

3.1 Background: Variational Autoencoders

We review variational autoencoders (VAEs) by describing a VAE for an input sequence $x_{1:T}$. When using a VAE, we assume a generative model that generates an input using a latent variable z, typically assumed to follow a multivariate Gaussian distribution. We seek to maximize the marginal likelihood of inputs $x_{1:T}$ when marginalizing out the latent variable z. Since this is typically intractable, especially when using continuous latent variables, we instead maximize a lower bound on the marginal log-likelihood (Kingma and Welling,



(a) VSL-G (b) VSL-GG-Flat (c) VSL-GG-Hier

Figure 1: Variational sequential labelers. The first row shows the original graphical models of each variant where shaded circles are observed variables. The second row shows how we perform inference and learning, showing inference models (in dashed lines), generative models (in solid lines), and classifier (in dotted lines). All models are trained to maximize $p_\theta(x_t|x_{-t})$ and predict the label l_t.

2014):

$$\log\ p_\theta(x_{1:T}) \geq \mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x_{1:T})}\left[\log p_\theta(x_{1:T}|z) - \log \frac{q_\phi(z|\ x_{1:T})}{p_\theta(z)}\right] = \underbrace{\mathop{\mathbb{E}}_{z\sim q_\phi(\cdot|x_{1:T})}[\log p_\theta(x_{1:T}|z)]}_{\text{Reconstruction Loss}} \underbrace{- KL(q_\phi(z|x_{1:T})\|p_\theta(z))}_{\text{KL divergence}} \quad (1)$$

where we have introduced the variational posterior q parametrized by new parameters ϕ. q is referred to as an "inference model" as it encodes an input into the latent space. We also have the generative model probabilities p parametrized by θ. The parameters are trained in a way that reflects a classical autoencoder framework: encode the input into a latent space, decode the latent space to reconstruct the input. These models are therefore referred to as "variational autoencoders".

The lower bound consists of two terms: reconstruction loss and KL divergence. The KL divergence term provides a regularizing effect during learning by ensuring that the learned posterior remains close to the prior over the latent variables.

3.2 Variational Sequential Labelers

We now introduce variational sequential labelers (VSLs) and propose several variants for sequence labeling tasks. Although the latent structure varies, a VSL maximizes the conditional probability of $p_\theta(x_t|x_{-t})$ and minimizes a classification loss using the latent variables as the input to the classifier. Unlike VAEs, VSLs do not autoencode the input, so they are more similar to recent conditional variational formulations (Sohn et al., 2015; Miao et al., 2016; Zhou and Neubig, 2017). Intuitively, the VSL variational objective is to find the information that is useful for predicting the word x_t from its surrounding context, which has similarities to objectives for learning word embeddings (Collobert et al., 2011; Mikolov et al., 2013). This objective serves as regularization for the labeled data and as an unsupervised objective for the unlabeled data.

All of our models use latent variables for each position in the sequence. These characteristics are shown in the visual depictions of our models in Figure 1. We consider variants with multiple latent variables per time step and attach the classifier to only particular variables. This causes the different latent variables to capture different characteristics.

In the following sections, we will describe various latent variable configurations that we will evaluate empirically in subsequent sections.

3.3 Single Latent Variable

We begin by defining a basic VSL and corresponding parametrization, which will also be used in other variants. This first model (which we call VSL-G and show in Figure 1a) has a Gaussian latent variable at each time step. VSL-G uses two training objectives; the first is similar to the lower bound on log-likelihood used by VAEs:

$$\log\ p_\theta(x_t|x_{-t}) \geq \mathop{\mathbb{E}}_{z_t\sim q_\phi(\cdot|\ x_{1:T},t)}[\log p_\theta(x_t|\ z_t) - \log \frac{q_\phi(z_t|\ x_{1:T},t)}{p_\theta(z_t|\ x_{-t})}] = \mathop{\mathbb{E}}_{z_t\sim q_\phi(\cdot|\ x_{1:T},t)}[\log p_\theta(x_t|\ z_t)] - KL(q_\phi(z_t|\ x_{1:T},t)\|\ p_\theta(z_t|\ x_{-t})) = U_0(x_{1:T},t) \quad (2)$$

VSL-G additionally uses a classifier f on the latent variable z_t which is trained with the following objective:

$$C_0(x_{1:T},l_t) = \mathop{\mathbb{E}}_{z_t\sim q_\phi(\cdot|x_{1:T},t)}[-\log f(l_t|z_t)] \quad (3)$$

The final loss is

$$L(x_{1:T},l_{1:T}) = \sum_{t=1}^{T}[C_0(x_{1:T},l_t) - \alpha U_0(x_{1:T},t)]$$

where α is a trade-off hyperparameter. α is set to zero during supervised training but it is tuned based on development set performance during semi-supervised training. The same procedure is adopted for the other VSL models below.

For the generative model, we parametrize $p_\theta(x_t|z_t)$ as a feedforward neural network with two hidden layers and ReLU (Nair and Hinton, 2010) as activation function. As reconstruction loss, we use cross-entropy over the words in the vocabulary. We defer the descriptions of the parametrization of $p_\theta(z_t|\ x_{-t})$ to Section 3.6.

We now discuss how we parametrize the inference model $q_\phi(z_t|x_{1:T}, t)$. We use a bidirectional gated recurrent unit (BiGRU; Chung et al., 2014) network to produce a hidden vector h_t at position t. The BiGRU is run over the input $x_{1:T}$, where each x_t is the concatenation of a word embedding and the concatenated final hidden states from a character-level BiGRU. The inference model $q_\phi(z_t|x_{1:T}, t)$ is then a single layer feedforward neural network that uses h_t as input. When parametrizing the posterior over latent variables in the following models below, we use this same procedure to produce hidden vectors with a BiGRU and then use them as input to feedforward networks. The structure of our inference model is similar to those used in previous state-of-the-art models for sequence labeling (Lample et al., 2016; Yang et al., 2017a).

In order to focus more on the effect of our variational objective, the classifier we use is always the same as our baseline model (see Section 4.3), which is a one layer feedforward neural network without a hidden layer, and it is also used in test-time prediction.

3.4 Flat Latent Variables

We next consider ways of factorizing the functionality of the latent variable into label-specific and other word-specific information. We introduce VSL-GG-Flat (shown in Figure 1b), which has two conditionally independent Gaussian latent variables at each time step, denote z_t and y_t for time step t. The variational lower bound is derived as follows:

$$\begin{aligned}
&\log\ p_\theta(x_t|x_{-t}) \geq \\
&\quad \mathop{\mathbb{E}}_{z_t,y_t\sim q_\phi(\cdot|x_{1:T},t)}[\log p_\theta(x_t|\ z_t, y_t) \\
&\quad - \log \frac{q_\phi(z_t|x_{1:T},t)}{p_\theta(z_t|x_{-t})} - \log \frac{q_\phi(y_t|x_{1:T},t)}{p_\theta(y_t|x_{-t})}] \\
&\quad = \mathop{\mathbb{E}}_{z_t,y_t\sim q_\phi(\cdot|x_{1:T},t)}[\log p_\theta(x_t|z_t, y_t)] \\
&\quad - KL(q_\phi(z_t|x_{1:T},t)\|p_\theta(z_t|x_{-t})) \\
&\quad - KL(q_\phi(y_t|x_{1:T},t)\|p_\theta(y_t|x_{-t})) \\
&\quad = U_1(x_{1:T}, t)
\end{aligned} \tag{4}$$

The classifier f is on the latent variable y_t and its loss is

$$C_1(x_{1:T}, l_t) = \mathop{\mathbb{E}}_{y_t\sim q_\phi(\cdot|x_{1:T},t)}[-\log f(l_t|y_t)] \tag{5}$$

The final loss for the model is

$$L(x_{1:T}, l_{1:T}) = \sum_{t=1}^{T}[C_1(x_{1:T}, l_t) - \alpha U_1(x_{1:T}, t)] \tag{6}$$

Where α is a trade-off hyperparameter.

Similarly to the VSL-G model, $q_\phi(z_t|x_{1:T}, t)$ and $q_\phi(y_t|x_{1:T}, t)$ are parametrized by single layer feedforward neural networks using the hidden state h_t as input.

3.5 Hierarchical Latent Variables

We also explore hierarchical relationships among the latent variables. In particular, we introduce the VSL-GG-Hier model which has two Gaussian latent variables with the hierarchical structure shown in Figure 1c. This model encodes the intuition that the word-specific latent information z_t may differ depending on the class-specific information of y_t.

For this model, the derivations are similar to Equations (4) and (5). The first is:

$$\begin{aligned}
&\log\ p_\theta(x_t|x_{-t}) \geq \\
&\quad \mathop{\mathbb{E}}_{z_t,y_t\sim q_\phi(\cdot|x_{1:T},t)}[\log p_\theta(x_t|z_t) \\
&\quad - \log \frac{q_\phi(z_t|y_t, x_{1:T},t)}{p_\theta(z_t|y_t, x_{-t})} - \log \frac{q_\phi(y_t|x_{1:T},t)}{p_\theta(y_t|x_{-t})}] \\
&\quad = \mathop{\mathbb{E}}_{z_t,y_t\sim q_\phi(\cdot|x_{1:T},t)}[\log p_\theta(x_t|z_t)] \\
&\quad - KL(q_\phi(z_t|y_t, x_{1:T},t)\|p_\theta(z_t|y_t, x_{-t})) \\
&\quad - KL(q_\phi(y_t|x_{1:T},t)\|p_\theta(y_t|x_{-t})) \\
&\quad = U_2(x_{1:T}, t)
\end{aligned} \tag{7}$$

The classifier f uses y_t as input and is trained with the following loss:

$$C_2(x_{1:T}, l_t) = \mathop{\mathbb{E}}_{y_t \sim q_\phi(\cdot|x_{1:T},t)} [-\log f(l_t|y_t)] \quad (8)$$

Note that C_1 and C_2 have the same form. The final loss is

$$L(x_{1:T}, l_{1:T}) = \sum_{t=1}^{T} [C_2(x_{1:T}, l_t) - \alpha\, U_2(x_{1:T}, t)] \quad (9)$$

Where α is a trade-off hyperparameter.

The hierarchical posterior $q_\phi(z_t|y_t, x_{1:T}, t)$ is parametrized by concatenating the hidden vector h_t and the random variable y_t and then using them as input to a single layer feedforward network.

3.6 Parametrization of Priors

Traditional variational models assume extremely simple priors (e.g., multivariate standard Gaussian distributions). Recently there have been efforts to learn the prior and posterior jointly during training (Fraccaro et al., 2016; Serban et al., 2017; Tomczak and Welling, 2018). In this paper, we follow this same idea but we do not explicitly parametrize the prior $p_\theta(z_t|x_{-t})$. This is partially due to the lack of computationally-efficient parametrization options for $p_\theta(z_t|x_{-t})$. In addition, since we are not seeking to do generation with our learned models, we can let part of the generative model be parametrized implicitly.

More specifically, the approach we use is to learn the priors by updating them iteratively. During training, we first initialize the priors of all examples as multivariate standard Gaussian distributions. As training proceeds, we use the last optimized posterior as our current prior based on a particular "update frequency" (see supplementary material for more details).

Our learned priors are implicitly modeled as

$$p_\theta^k(z_t|x_{-t}) \approx \sum_x q_\phi^{k-1}(z_t|X_t = x, x_{-t}, t) p_{\text{data}}(X_t = x|x_{-t}) \quad (10)$$

where p_{data} is the empirical data distribution, X_t is a random variable corresponding to the observation at position t, and k is the prior update time step. The intuition here is that the prior is obtained by marginalizing over values for the missing observation represented by the random variable X_t. The posterior q_ϕ^{k-1} is as defined in our latent variable models. We assume $p_{\text{data}}(X_t = x|x_{-t}) = 0$ for $x_{1:T} \notin$ training set. For context x_{-t} that can pair with multiple values of X_t, its prior is the data-dependent weighted average posterior. For simplicity of implementation and efficient computation, however, if context x_{-t} can pair with multiple values in our training data, we ignore this fact and simply use instance-dependent posteriors. Another way to view this is as conditioning on the index of the training examples while parametrizing the above. That is

$$p_\theta^{k,i}(z_t|x_{-t}) \leftarrow q_\phi^{k-1,i}(z_t|x_{1:T}, t) \quad (11)$$

where i is the index of the instance.

3.7 Training

In this subsection, we introduce techniques we have used to address difficulties during training.

Reparametrization Trick. It is challenging to use gradient descent for a random variable as it involves a non-differentiable sampling procedure. Kingma and Welling (2014) introduced a reparametrization trick to tackle this problem. They parametrize a Gaussian random variable z as $u_\varphi(x) + g_\psi(x) \circ \epsilon$ where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ and $u_\varphi(x)$, $g_\psi(x)$ are deterministic and differentiable functions, so the gradient can go through $u_\varphi(\cdot)$ and $g_\psi(\cdot)$. In our experiments, we use one sample for each time step during training. For evaluation at test time, we use the mean value $u_\varphi(x)$.

KL Divergence Weight Annealing. Although the use of prior updating lets us avoid tuning the weight of the KL divergence, the simple priors can still hinder learning during the initial stages of training. To address this, we follow the method described by Bowman et al. (2016) to add weights to all KL divergence terms and anneal the weights from a small value to 1.

4 Experiments

We describe key details of our experimental setup in the subsections below but defer details about hyperparameter tuning to the supplementary material. Our implementation is available at `https://github.com/mingdachen/vsl`

4.1 Datasets

We evaluate our model on the CoNLL 2003 English NER dataset (Tjong Kim Sang and De Meulder, 2003) and 7 POS tagging datasets: the

Twitter tagging dataset of Gimpel et al. (2011) and Owoputi et al. (2013), and 6 languages from the Universal Dependencies (UD) 1.4 dataset (McDonald et al., 2013).

Twitter POS Dataset. The Twitter dataset has 25 tags. We use OCT27TRAIN and OCT27DEV as the training set, OCT27TEST as the development set, and DAILY547 as the test set. We randomly sample {1k, 2k, 3k, 4k, 5k, 10k, 20k, 30k, 60k} tweets from 56 million English tweets as our unlabeled data and tune the amount of unlabeled data based on development set accuracy.

UD POS Datasets. The UD datasets have 17 tags. We use French, German, Spanish, Russian, Indonesian and Croatian. We follow the same setup as Zhang et al. (2017), randomly sampling 20% of the original training set as our labeled data and 50% as unlabeled data. There is no overlap between the labeled and unlabeled data. See Zhang et al. (2017) for more details about the setup.

NER Dataset. We use the BIOES labeling scheme and report micro-averaged F_1. We preprocessed the text by replacing all digits with 0. We randomly sample 10% of the original training set as our labeled data and 50% as unlabeled data. We also ensure there is no overlap between the labeled and unlabeled data.

4.2 Pretrained Word Embeddings

For all experiments, we use pretrained 100-dimensional word embeddings. For Twitter, we trained skip-gram embeddings (Mikolov et al., 2013) on a dataset of 56 million English tweets. For the UD datasets, we trained skip-gram embeddings on Wikipedia for each of the six languages. For NER, we use 100-dimensional pretrained GloVe (Pennington et al., 2014) embeddings. Our models perform better with word embeddings kept fixed during training while for the baselines the word embeddings are fine tuned as this improves the baseline performance.

4.3 Baselines

Our primary baseline is a BiGRU tagger where the input consists of the concatenation of a word embedding and the concatenation of the final hidden states of a character-level BiGRU. This BiGRU architecture is identical to that used in the inference networks in our VSL models. Predictions are made based on a linear transformation given the current hidden state. The output dimensionality of the transformation is task-dependent (e.g., 25 for Twitter tagging). We use the standard per-position cross entropy loss for training.

We also report results from the best systems from Zhang et al. (2017), namely the NCRF and NCRF-AE models. Both use feedforward networks as encoders and conditional random field layers for capturing sequential information. The NCRF-AE model additionally can benefit from unlabeled data.

	dev.		test	
	acc.	ULΔ	acc.	ULΔ
BiGRU baseline	90.8	-	90.6	-
VSL-G	91.1	+0.1	-	-
VSL-GG-Flat	91.4	+0.1	-	-
VSL-GG-Hier	***91.6***	***+0.3***	***91.6***	*+0.3*

(a) Twitter tagging accuracies (%)

	dev.		test	
	F_1	ULΔ	F_1	ULΔ
BiGRU baseline	87.6	-	83.7	-
VSL-G	87.8	+0.1	-	-
VSL-GG-Flat	88.0	+0.1	-	-
VSL-GG-Hier	***88.4***	***+0.2***	***84.7***	*+0.0*

(b) NER F_1 score (%)

Table 1: For dev and test, we show results when only using labeled data and the change in performances ("ULΔ") when adding unlabeled data. Bold is highest in each column. Italic is the best model including unlabeled data. We only show test results for the baseline and our best-performing model, which achieves 91.9% accuracy on the Twitter test set and 84.7% F_1 on the NER test set when using unlabeled data.

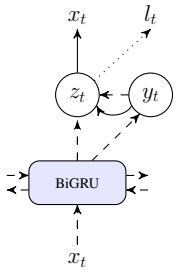
5 Results

Table 1a shows results on the Twitter development and test sets. All of our VSL models outperform the baseline and our best VSL models outperform the BiGRU baseline by 0.8–1% absolute. When comparing different latent variable configurations, we find that a hierarchical structure performs best. Without unlabeled data, our models already outperform the BiGRU baseline. Adding unlabeled data enlarges the gap between the baseline and our models by up to 0.1–0.3% absolute.

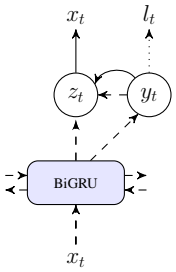
Table 1b shows results on the CoNLL 2003 NER development and test sets. We observe similar trends as in the Twitter data, except that the model does not show improvement on the test set when adding unlabeled data.

	French		German		Indonesian		Spanish		Russian		Croatian	
	acc.	ULΔ	acc.	ULΔ	acc.	ULΔ	acc.	ULΔ	acc.	ULΔ	acc.	ULΔ
NCRF	93.4	-	90.4	-	88.4	-	91.2	-	86.6	-	86.1	-
NCRF-AE	93.7	+0.2	90.8	+0.2	89.1	+0.3	91.7	+0.5	87.8	+1.1	87.9	+1.2
BiGRU baseline	95.9	-	92.6	-	92.2	-	94.7	-	95.2	-	95.6	-
VSL-G	96.1	+0.0	92.8	+0.0	92.3	+0.0	94.8	+0.1	95.3	+0.0	95.6	+0.1
VSL-GG-Flat	96.1	+0.0	93.0	**+0.1**	92.4	**+0.1**	95.0	+0.1	95.5	**+0.1**	95.8	+0.1
VSL-GG-Hier	***96.4***	***+0.1***	***93.3***	***+0.1***	***92.8***	***+0.1***	***95.3***	***+0.2***	***95.9***	***+0.1***	***96.3***	***+0.2***

Table 2: Tagging accuracies (%) on UD test sets. For each language, we show test accuracy ("acc.") when only using labeled data and the change in test accuracy ("ULΔ") when adding unlabeled data. Results for NCRF and NCRF-AE are from Zhang et al. (2017), though results are not strictly comparable because we used pretrained word embeddings for all languages on Wikipedia. Bold is highest in each column, excluding the NCRF variants. Italic is the best accuracy including the unlabeled data.



(a) VSL-GG-Hier with classification loss on z



(b) VSL-GG-Hier

Figure 2: Comparison of attaching classification loss to different latent variables in VSL-GG-Hier.

Table 2 shows our results on the UD datasets. The trends are broadly consistent with those of Table 1a and 1b. The best performing models use hierarchical structure in the latent variables. There are some differences across languages. For French, German, Indonesian and Russian, VSL-G does not show improvement when using unlabeled data. This may be resolved with better tuning, since the model actually shows improvement on the dev set.

Note that results reported by Zhang et al. (2017) and ours are not strictly comparable as their word embeddings were only pretrained on the UD training sets while ours were pretrained on Wikipedia. Nonetheless, they also mentioned that using embeddings pretrained on larger unlabeled data did not help. We include these results to show that our baselines are indeed strong compared to prior results reported in the literature.

	Twitter		NER		UD average	
	acc.	ULΔ	F_1	ULΔ	acc.	ULΔ
classifier on y	91.6	+0.3	88.4	+0.2	95.0	+0.1
classifier on z	91.1	+0.2	87.8	+0.1	94.4	+0.0

Table 3: Twitter and NER dev results (%), UD averaged test accuracies (%) for two choices of attaching the classification loss to latent variables in the VSL-GG-Hier model. All previous results for VSL-GG-Hier used the classification loss on y.

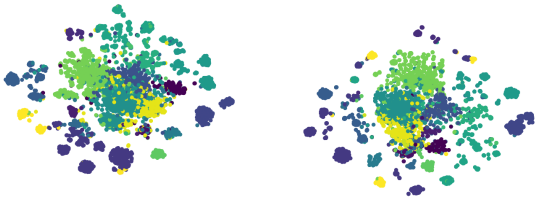
6 Discussion

6.1 Effect of Position of Classification Loss

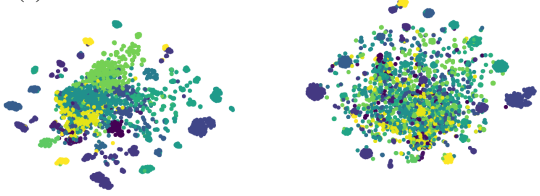
We investigate the effect of attaching the classifier to different latent variables. In particular, for the VSL-GG-Hier model, we compare the attachment of the classifier between z and y. See Figure 2. The results in Table 3 suggest that attaching the reconstruction and classification losses to the same latent variable (z) harms accuracy although attaching the classifier to z effectively gives the classifier an extra layer. We can observe why this occurs by looking at the latent variable visualizations in Figure 3d. Compared with Figure 3e, where the two variables are more clearly disentangled, the latent variables in Figure 3d appear to be capturing highly similar information.

6.2 Effect of Latent Hierarchy

To verify our assumption of the latent structure, we visualize the latent space for Gaussian models using t-SNE (Maaten and Hinton, 2008) in Figure 3. The BiGRU baseline (Figure 3a) and the VSL-G (Figure 3b) do not show significant differences. However, when using multiple latent variables, the different latent variables capture different characteristics. In the VSL-GG-Flat model (Figure 3c), the y variable (the upper plot) reflects

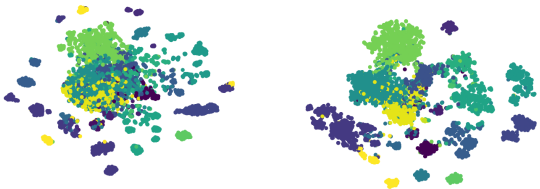


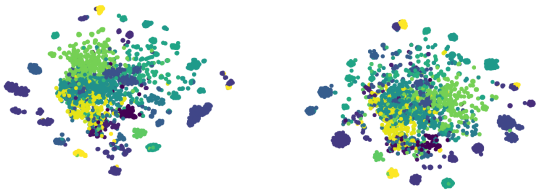
(a) BiGRU Baseline



(b) VSL-G

(c) VSL-GG-Flat





(d) VSL-GG-Hier, classification loss on z

(e) VSL-GG-Hier

Figure 3: t-SNE visualization of Gaussian latent variables and baseline hidden states for Twitter development set. In plot 3c, 3d, and 3e, the upper subplot is latent variable y and the lower is z. Each point in the plot is a token and the color represents the true tag of the token.

the clustering of the tagging space much more closely than the z variable (the lower plot). Since both variables are used to reconstruct the word, but only the y variable is trained to predict the tag, it appears that z is capturing other information useful for reconstructing the word. However, since they are both used for reconstruction, the two spaces show signs of alignment; that is, the "tag" latent variable y does not show as clean a separation into tag clusters as the y variable in the VSL-GG-Hier model in Figure 3e.

In Figure 3e (VSL-GG-Hier), the clustering of words with respect to the tag is clearest. This may account for the consistently better performance of this model relative to the others. The z variable reflects a space that is conditioned on y but that diverges from it, presumably in order to better reconstruct the word. The closer the latent variable is to the decoder output, the weaker the tagging information becomes while other word-specific information becomes more salient.

Figure 3d shows that VSL-GG-Hier with classification loss on z, which consistently underperforms both the VSL-GG-Flat and VSL-GG-Hier models in our experiments, appears to be capturing the same latent space in both variables. Since the z variable is used to both predict the tag and reconstruct the word, it must capture both the tag and word reconstruction spaces, and may be limited by capacity in doing so. The y variable does not seem to be contributing much modeling power, as its space is closely aligned to that of z.

	Twitter		NER	
	acc.	no VR	F_1	no VR
BiGRU baseline	90.8	-	87.6	-
VSL-G	91.1	90.9	87.8	87.7
VSL-GG-Flat	91.4	90.9	88.0	87.8
VSL-GG-Hier	91.6	91.0	88.4	87.9

Table 4: Results on Twitter and NER dev sets. For each model, we show supervised results for the models with variational regularization ("acc." or F_1) and results when replacing variational components with their deterministic counterparts ("no VR").

6.3 Effect of Variational Regularization

We investigate the beneficial effects of variational frameworks ("variational regularization") by replacing our variational components in VSLs with their deterministic counterparts, which do not have randomness in the latent space and do not use the KL divergence term during optimization. Note that these BiGRU encoders share the same architectures as their variational posterior counterparts and still use both the classification and reconstruction losses. While other subsets of losses could be considered in this comparison, our motivation is to compare two settings that correspond to well-known frameworks. The "no VR" setting corresponds roughly to the combination of a classifier and a traditional autoencoder. We note that these experiments do not use any unlabeled data.

The results in Table 4 demonstrate that compared to the baseline BiGRU, adding the reconstruction loss ("VSL-G, no VR") yields only 0.1 improvement for both Twitter and NER. Although adding hierarchical structure further improves performance, the improvements are small (+0.1 and +0.2 for Twitter and NER respectively). For VSL-

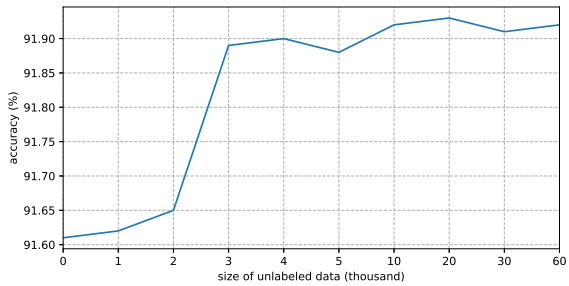


Figure 4: Twitter dev accuracies (%) when varying the amount of unlabeled data.

GG-Hier, variational regularization accounts for relatively large differences of 0.6 for Twitter and 0.5 for NER. These results show that the improvements do not come solely from adding a reconstruction objective to the learning procedure. In limited preliminary experiments, we did not find a benefit from adding unlabeled data under the "no VR" setting.

6.4 Effect of Unlabeled Data

In order to examine the effect of unlabeled data, we report our Twitter dev accuracies when varying the unlabeled data size. We choose VSL-GG-Hier as the model for this experiment since it benefits the most from unlabeled data. As Figure 4 shows, gradually adding unlabeled data helps a little at the beginning. Further adding unlabeled data boosts the accuracy of the model. The improvements that come from unlabeled data quickly plateau after the amount of unlabeled data goes beyond 10,000. This suggests that with little unlabeled data, the model is incapable of fully utilizing the information in the unlabeled data. However if the amount of unlabeled data is too large, the supervised training signal becomes too weak to extract something useful from the unlabeled data.

We also notice that when there is a large amount of unlabeled data, it is always better to pretrain the prior first using a small α (e.g., 0.1) and then use it as a warm start to train a new model using a larger α (e.g., 1.0). Tuning the weight of the KL divergence could achieve a similar effect, but it may require tuning the weight for labeled data and unlabeled data separately. We prefer to pretrain the prior as it is simpler and involves less hyperparameter tuning.

7 Conclusion

We introduced variational sequential labelers for semi-supervised sequence labeling. They consist of latent-variable generative models with flexible parametrizations for the variational posterior (using RNNs over the entire input sequence) and a classifier at each time step. Our best models use multiple latent variables arranged in a hierarchical structure. We demonstrate systematic improvements in NER and POS tagging accuracy across 8 datasets over a strong baseline. We also find small, but consistent, improvements by using unlabeled data.

Acknowledgments

We would like to thank NVIDIA for donating GPUs used in this research, the anonymous reviewers for their comments that improved this paper, and Google for a faculty research award to K. Gimpel that partially supported this research. This research was funded by NSF grant 1433485.

References

Y. Altun, D. McAllester, and M. Belkin. 2006. Maximum margin semi-supervised learning for structured variables. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 33–40. MIT Press.

Isabelle Augenstein and Anders Søgaard. 2017. Multi-task learning of keyphrase boundary classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 341–346. Association for Computational Linguistics.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 164–169. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21. Association for Computational Linguistics.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2980–2988. Curran Associates, Inc.

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.

Marco Fraccaro, Søren Kaae Sø nderby, Ulrich Paquet, and Ole Winther. 2016. Sequential neural models with stochastic layers. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2199–2207. Curran Associates, Inc.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 42–47. Association for Computational Linguistics.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. 2015. Draw: A recurrent neural network for image generation. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1462–1471, Lille, France. PMLR.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1587–1596, International Convention Centre, Sydney, Australia. PMLR.

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

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3581–3589. Curran Associates, Inc.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*.

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.

Lars Maale, Casper Kaae Snderby, Sren Kaae Snderby, and Ole Winther. 2016. Auxiliary deep generative models. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1445–1453, New York, New York, USA. PMLR.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Gideon S. Mann and Andrew McCallum. 2008. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *Proceedings of ACL-08: HLT*, pages 870–878. Association for Computational Linguistics.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97. Association for Computational Linguistics.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1727–1736, New York, New York, USA. PMLR.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL 2004: Main Proceedings*, pages 337–342, Boston, Massachusetts, USA. Association for Computational Linguistics.

Andriy Mnih and Karol Gregor. 2014. Neural variational inference and learning in belief networks. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1791–1799, Bejing, China. PMLR.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–390. Association for Computational Linguistics.

Yookoon Park, Jaemin Cho, and Gunhee Kim. 2018. A hierarchical latent structure for variational conversation modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1792–1801. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765. Association for Computational Linguistics.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418. Association for Computational Linguistics.

Ariadna Quattoni, Sybor Wang, Louis-Philippe Morency, Morency Collins, and Trevor Darrell. 2007. Hidden conditional random fields. *IEEE transactions on pattern analysis and machine intelligence*, 29(10).

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2121–2130. Association for Computational Linguistics.

Danilo Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France. PMLR.

Iulian Vlad Serban, Alexander G. Ororbia, Joelle Pineau, and Aaron Courville. 2017. Piecewise latent variables for neural variational text processing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 422–432. Association for Computational Linguistics.

Anders Søgaard. 2011. Semi-supervised condensed nearest neighbor for part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 48–52, Portland, Oregon, USA. Association for Computational Linguistics.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3483–3491. Curran Associates, Inc.

Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 167–176, Cambridge, MA. Association for Computational Linguistics.

Xu Sun and Jun'ichi Tsujii. 2009. Sequential labeling with latent variables: An exact inference algorithm and its efficient approximation. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 772–780. Association for Computational Linguistics.

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-Volume 4*, pages 142–147. Association for Computational Linguistics.

Jakub Tomczak and Max Welling. 2018. Vae with a vampprior. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1214–1223, Playa Blanca, Lanzarote, Canary Islands. PMLR.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. 2017a. Transfer learning for sequence tagging with hierarchical recurrent networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017b. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3881–3890, International Convention Centre, Sydney, Australia. PMLR.

Biao Zhang, Deyi Xiong, jinsong su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530. Association for Computational Linguistics.

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

Chunting Zhou and Graham Neubig. 2017. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 310–320. Association for Computational Linguistics.