A Multi-Task Approach for Disentangling Syntax and Semantics in Sentence Representations

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

Abstract

We propose a generative model for a sentence that uses two latent variables, with one intended to represent the syntax of the sentence and the other to represent its semantics. We show we can achieve better disentanglement between semantic and syntactic representations by training with multiple losses, including losses that exploit aligned paraphrastic sentences and word-order information. We also investigate the effect of moving from bag-of-words to recurrent neural network modules. We evaluate our models as well as several popular pretrained embeddings on standard semantic similarity tasks and novel syntactic similarity tasks. Empirically, we find that the model with the best performing syntactic and semantic representations also gives rise to the most disentangled representations.[1]

1 Introduction

As generative latent variable models, especially of the continuous variety (Kingma and Welling, 2014; Goodfellow et al., 2014), have become increasingly important in natural language processing (Bowman et al., 2016; Gulrajani et al., 2017), there has been increased interest in learning models where the latent representations are disentangled (Hu et al., 2017). Much of the recent NLP work on learning disentangled representations of text has focused on disentangling the representation of attributes such as sentiment from the representation of content, typically in an effort to better control text generation (Shen et al., 2017; Zhao et al., 2017; Fu et al., 2018).

In this work, we instead focus on learning sentence representations that disentangle the syntax and the semantics of a sentence. We are moreover interested in disentangling these representations not for the purpose of controlling generation, but for the purpose of calculating semantic or syntactic similarity between sentences (but not both). To this end, we propose a generative model of a sentence which makes use of both semantic and syntactic latent variables, and we evaluate the induced representations on both standard semantic similarity tasks and on several novel syntactic similarity tasks.

We use a deep generative model consisting of von Mises Fisher (vMF) and Gaussian priors on the semantic and syntactic latent variables (respectively) and a deep bag-of-words decoder that conditions on these latent variables. Following much recent work, we learn this model by optimizing the ELBO with a VAE-like (Kingma and Welling, 2014; Rezende et al., 2014) approach.

Our learned semantic representations are evaluated on the SemEval semantic textual similarity (STS) tasks (Agirre et al., 2012; Cer et al., 2017). Because there has been less work on evaluating syntactic representations of sentences, we propose several new syntactic evaluation tasks, which involve predicting the syntactic analysis of an unseen sentence to be the syntactic analysis of its nearest neighbor (as determined by the latent syntactic representation) in a large set of annotated sentences.

In order to improve the quality and disentanglement of the learned representations, we incorporate simple additional losses in our training, which are designed to force the latent representations to capture different information. In particular, our semantic multi-task losses make use of aligned paraphrase data, whereas our syntactic multi-task loss makes use of word-order information. Additionally, we explore different encoder and decoder architectures for learning better syntactic representations. Experimentally, we find that by training in this way we are able to force the learned represen-

[1]Code and data are available at github.com/mingdachen/disentangle-semantics-syntax

tations to capture different information (as measured by the performance gap between the latent representations on each task). Moreover, we find that we achieve the best performance on all tasks when the learned representations are most disentangled.

2 Related Work

There is a growing amount of work on learning interpretable or disentangled latent representations both in machine learning (Tenenbaum and Freeman, 2000; Reed et al., 2014; Makhzani et al., 2015; Mathieu et al., 2016; Higgins et al., 2016; Chen et al., 2016; Hsu et al., 2017) and in various NLP applications, including sentence sentiment and style transfer (Hu et al., 2017; Shen et al., 2017; Fu et al., 2018; Zhao et al., 2018, *inter alia*), morphological reinflection (Zhou and Neubig, 2017), semantic parsing (Yin et al., 2018), text generation (Wiseman et al., 2018), and sequence labeling (Chen et al., 2018). Another related thread of work is text-based variational autoencoders (Miao et al., 2016; Bowman et al., 2016; Serban et al., 2017; Xu and Durrett, 2018).

In terms of syntax and semantics in particular, there is a rich history of work in analyzing their interplay in sentences (Jurafsky, 1988; van Valin, Jr., 2005). We do not intend to claim that the two can be entirely disentangled in distinct representations. Rather, our goal is to propose modica of knowledge via particular multi-task losses and measure the extent to which this knowledge leads learned representations to favor syntactic or semantic information from a sentence.

There has been prior work with similar goals for representations of words (Mitchell and Steedman, 2015) and bilexical dependencies (Mitchell, 2016), finding that decomposing syntactic and semantic information can lead to improved performance on semantic tasks. We find similar trends in our results, but at the level of sentence representations. A similar idea has been explored for text generation (Iyyer et al., 2018), where adversarial examples are generated by controlling syntax.

Some of our losses use sentential paraphrases, relating them to work in paraphrase modeling (Wieting et al., 2016; Wieting and Gimpel, 2018). Deudon (2018) recently proposed a variational framework for modeling paraphrastic sentences, but our focus here is on learning disentangled representations.

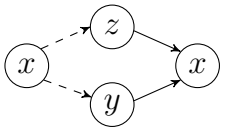

Figure 1: Graphical model of VGVAE. Dashed lines indicate inference model. Solid lines indicate generative model.

As part of our evaluation, we develop novel syntactic similarity tasks for sentence representations learned without any syntactic supervision. These evaluations relate to the broad range of work in unsupervised parsing (Klein and Manning, 2004) and part-of-speech tagging (Christodoulopoulos et al., 2010). However, our evaluations differ from previous evaluations in that we employ k-nearest-neighbor syntactic analyzers using our syntactic representations to choose nearest neighbors.

There is a great deal of work on applying multi-task learning to various NLP tasks (Plank et al., 2016; Rei, 2017; Augenstein and Søgaard, 2017; Bollmann et al., 2018, *inter alia*) and, recently, as a way of improving the quality or disentanglement of learned representations (Zhao et al., 2017; Goyal et al., 2017; Du et al., 2018; John et al., 2018).

3 Proposed Approach

Our goal is to extract the disentangled semantic and syntactic information from sentence representations. To achieve this, we introduce the vMF-Gaussian Variational Autoencoder (VGVAE). As shown in Figure 1, VGVAE assumes a sentence is generated by conditioning on two independent variables: semantic variable y and syntactic variable z. In particular, our model gives rise to the following joint likelihood

$$\begin{aligned} p_\theta(x, y, z) &= p_\theta(y)p_\theta(z)p_\theta(x|y, z) \\ &= p_\theta(y)p_\theta(z)\prod_{t=1}^{T} p(x_t \mid y, z), \end{aligned}$$

where x_t is the tth word of x, T is the sentence length, and $p(x_t|y, z)$ is given by a softmax over a vocabulary of size V. Further details on the parameterization are given below.

To perform inference, we assume a factored posterior $q_\phi(y, z|x) = q_\phi(y|x)q_\phi(z|x)$, as has been used in prior work (Zhou and Neubig, 2017; Chen et al., 2018). Learning of VGVAE maximizes a lower bound on marginal log-likelihood:

$$
\begin{aligned}
&\log p_\theta(x) \geq \mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}[\log p_\theta(x|z,y) \\
&- \log \frac{q_\phi(z|x)}{p_\theta(z)} - \log \frac{q_\phi(y|x)}{p_\theta(y)}] \\
&= \mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}[\log p_\theta(x|z,y)] - KL(q_\phi(z|x)\|p_\theta(z)) \\
&- KL(q_\phi(y|x)\|p_\theta(y)) \stackrel{\text{def}}{=} \text{ELBO}
\end{aligned} \tag{1}
$$

3.1 Parameterizations

VGVAE uses two distribution families in defining the posterior over latent variables, namely, the von Mises-Fisher (vMF) distribution and the Gaussian distribution.

vMF Distribution. vMF can be regarded as a Gaussian distribution on a hypersphere with two parameters: μ and κ. $\mu \in \mathbb{R}^m$ is a normalized vector (i.e. $\|\mu\|_2 = 1$) defining the mean direction. $\kappa \in \mathbb{R}_{\geq 0}$ is often referred to as a concentration parameter analogous to the variance in a Gaussian distribution. vMF has been used for modeling similarity between two sentences (Guu et al., 2018), which is particularly suited to our purpose here, since we will evaluate our semantic representations in the context of modeling paraphrases (See Sections 4.1 and 4.2 for more details). Therefore, we assume $q_\phi(y|x)$ follows vMF$(\mu_\alpha(x), \kappa_\alpha(x))$ and the prior $p_\theta(y)$ follows the uniform distribution vMF$(\cdot, 0)$.

With this choice of prior and posterior distribution, the $KL(q_\phi(y|x)\|p_\theta(y))$ appearing in the ELBO can be computed in closed-form:

$$
\begin{aligned}
&\kappa_\alpha \frac{\mathcal{I}_{m/2}(\kappa_\alpha)}{\mathcal{I}_{m/2-1}(\kappa_\alpha)} + (m/2-1)\log\kappa_\alpha - \\
&(m/2)\log(2\pi) - \log \mathcal{I}_{m/2-1}(\kappa_\alpha) + \\
&\frac{m}{2}\log\pi + \log 2 - \log\Gamma(\frac{m}{2}),
\end{aligned} \tag{2}
$$

where $\mathcal{I}_v$ is the modified Bessel function of the first kind at order v and $\Gamma(\cdot)$ is the Gamma function. We follow Davidson et al. (2018) and use an acceptance-rejection scheme to sample from vMF.

Gaussian Distribution.[2] We assume $q_\phi(z|x)$ follows a Gaussian distribution $\mathcal{N}(\mu_\beta(x), \text{diag}(\sigma_\beta(x)))$ and that the prior $p_\theta(z)$ is $\mathcal{N}(0, I_d)$, where I_d is an $d \times d$ identity matrix. Since we only consider a diagonal covariance matrix, the KL divergence term $KL(q_\phi(z|x)\|p_\theta(z))$ can also be computed efficiently:

$$
\frac{1}{2}(-\sum_i \log\sigma_{\beta i} + \sum_i \sigma_{\beta i} + \sum_i \mu_{\beta i}^2 - d) \tag{3}
$$

Inference and Generative Models. The inference models $q_\phi(y|x)$ and $q_\phi(z|x)$ are two independent word averaging encoders with additional linear feedforward neural networks for producing $\mu(x)$ and $\sigma(x)$ (or $\kappa(x)$). The generative model $p_\theta(x|y,z)$ is a feedforward neural network g_θ with the output being a bag of words. In particular, the expected output log-probability (the first term in Eq. 1) is computed as follows:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}[\log p_\theta(x|y,z)] = \\
&\mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}\left[\sum_{t=1}^{T}\log\frac{\exp g_\theta([y;z])_{x_t}}{\sum_{j=1}^{V}\exp g_\theta([y;z])_j}\right]
\end{aligned}
$$

Where V is the vocabulary size, $[;]$ indicates concatenation, T is the sentence length and x_t is the index of the t'th word's word type.

Recurrent Neural Networks. To facilitate better learning of syntax, we also consider replacing both the generative and inference models with RNN-based sequence models, rather than bag-of-words models. In this setting, the generative model $p_\theta(x|y,z)$ is a unidirectional long-short term memory network (LSTM; Hochreiter and Schmidhuber, 1997) and a linear feedforward neural network for predicting the word tokens (shown in Figure 2). The expected output log-probability is computed as follows:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}[\log p_\theta(x|y,z)] = \\
&\mathop{\mathbb{E}}_{\substack{y\sim q_\phi(y|x)\\ z\sim q_\phi(z|x)}}\left[\sum_{t=1}^{T}\log p_\theta(x_t|y,z,x_{1:t-1})\right]
\end{aligned}
$$

Where V is the vocabulary size, T is the sentence length and x_t is the index of the t'th word's word type.

The inference model $q_\phi(y|x)$ is still a word averaging encoder, but $q_\phi(z|x)$ is parameterized by

[2] In preliminary experiments, we observed that using two distribution families can lead to better performance. This is presumably because the Gaussian distribution complements the norm information lost in the vMF distribution.

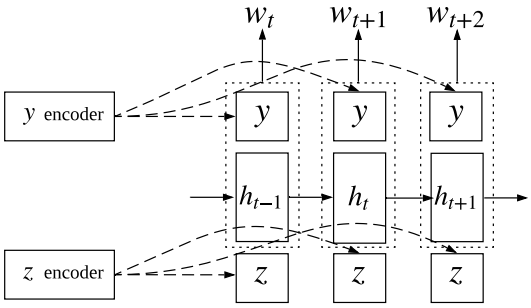

Figure 2: Diagram showing LSTM decoder that uses the semantic variable y and the syntactic variable z.

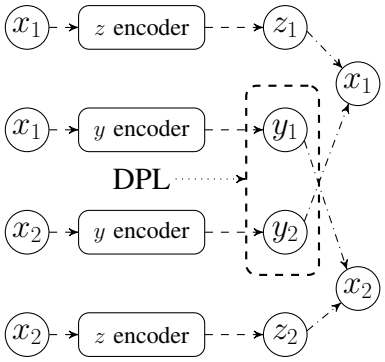

Figure 3: Diagram showing the training process when using the discriminative paraphrase loss (DPL; dotted lines) and paraphrase reconstruction loss (PRL; dash-dotted lines). The pair (x_1, x_2) is a sentential paraphrase pair, the y's are the semantic variables corresponding to each x, and the z's are syntactic variables.

a bidirectional LSTM, where we concatenate the forward and backward hidden states and then take the average. The output of the LSTM is then used as input to a feedforward network with one hidden layer for producing $\mu(x)$ and $\sigma(x)$ (or $\kappa(x)$).

In the following sections, we will introduce several losses that will be added into the training of our base model, which empirically shows the ability of further disentangling the functionality between the semantic variable y and the syntactic variable z.

4 Multi-Task Training

We attempt to improve the quality and disentanglement of our semantic and syntactic representations by introducing additional losses, which encourage y to capture semantic information and z to capture syntactic information. We elaborate on these losses below.

4.1 Paraphrase Reconstruction Loss

Our first loss is a paraphrase reconstruction loss (PRL). The key assumption underlying the PRL is that for a paraphrase pair x_1, x_2, the semantic information is equivalent between the two sentences and only the syntactic information varies. To impose such constraints, PRL is defined as

$$\begin{aligned} &\mathop{\mathbb{E}}_{\substack{y_2\sim q_\phi(y|x_2)\\ z_1\sim q_\phi(z|x_1)}}[-\log p_\theta(x_1|y_2,z_1)]+ \\ &\mathop{\mathbb{E}}_{\substack{y_1\sim q_\phi(y|x_1)\\ z_2\sim q_\phi(z|x_2)}}[-\log p_\theta(x_2|y_1,z_2)] \end{aligned} \tag{4}$$

That is, we swap the semantic variables, keep the syntactic variables, and attempt to reconstruct the sentences (shown in Figure 3). While instead of using a multi-task objective we could directly model paraphrases x_1 and x_2 as being generated by the same y (which naturally suggests a product-of-experts style posterior, as in Wu and Goodman (2018)), we found that for the purposes of our downstream tasks training with the multi-task loss gave superior results.

4.2 Discriminative Paraphrase Loss

Our second loss is a discriminative paraphrase loss (DPL). The DPL explicitly encourages the similarity of paraphrases x_1, x_2 to be scored higher than the dissimilar sentences n_1, n_2 (i.e., negative samples; see Sec. 5 for more details) by a given margin δ. As shown in Figure 3, the similarity function in this loss only uses the semantic variables in the sentences. The loss is defined as

$$\begin{aligned} &\max(0, \delta - d(x_1, x_2) + d(x_1, n_1))+ \\ &\max(0, \delta - d(x_1, x_2) + d(x_2, n_2)) \end{aligned} \tag{5}$$

The similarity function we choose is the cosine similarity between the mean directions of the semantic variables from the two sentences:

$$d(x_1, x_2) = \text{cosine}(\mu_\alpha(x_1), \mu_\alpha(x_2)) \tag{6}$$

4.3 Word Position Loss

It has been observed in previous work that word order typically contributes little to the modelling of semantic similarity (Wieting et al., 2016). We interpret this as evidence that word position information is more relevant to syntax than semantics, at least as evaluated by STS tasks. To guide the syntactic variable to represent word order, we introduce a word position loss (WPL). Although our

word averaging encoders only have access to the bag of words of the input, using this loss can be viewed as a denoising autoencoder where we have maximal input noise (i.e., an orderless representation of the input) and the encoders need to learn to reconstruct the ordering. For both word averaging encoders and LSTM encoders, WPL is parameterized by a three-layer feedforward neural network $f(\cdot)$ with input from the concatenation of the samples of the syntactic variable z and the embedding vector e_i at input position i; we then attempt to predict a one-hot vector representing the position i. More specifically, we define

$$\text{WPL} \stackrel{\text{def}}{=\!=} \mathop{\mathbb{E}}_{z\sim q_\phi(z|x)} \left[-\sum_i \log \text{softmax}(f([e_i; z]))_i \right]$$

where $\text{softmax}(\cdot)_i$ indicates the probability at position i.

5 Training

KL Weight. Following previous work on VAEs (Higgins et al., 2016; Alemi et al., 2016), we attach a weight to the KL divergence and tune it based on development set performance.

Negative Samples. When applying DPL, we select negative samples based on maximizing cosine similarity to sentences from a subset of the data. In particular, we accumulate k mini-batches during training, yielding a "mega-batch" $\mathcal{S}$ (Wieting and Gimpel, 2018). Then the negative samples are selected based on the following criterion:

$$n_1 = \mathop{\text{argmax}}_{n\in\mathcal{S}\wedge n\neq x_2} \text{cosine}(\mu_\alpha(x_1), \mu_\alpha(n))$$

where x_1, x_2 forms the paraphrase pair and the mega-batch size is fixed to $k = 20$ for all of our experiments. Since all of our models are trained from scratch, we observed some instabilities with DPL during the initial stages of training. We suspect that this is because the negative samples at these initial stages are of low quality. To overcome this issue, DPL is included starting at the second epoch of training so that the models can have a warm start.

6 Experiments

6.1 Setup

We subsampled half a million paraphrase pairs from ParaNMT-50M (Wieting and Gimpel, 2018) as our training set. We use SemEval semantic textual similarity (STS) task 2017 (Cer et al., 2017) as a development set. For semantic similarity evaluation, we use the STS tasks from 2012 to 2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016) and the STS benchmark test set (Cer et al., 2017). For evaluating syntactic similarity, we propose several evaluations. One uses the gold parse trees from the Penn Treebank (Marcus et al., 1993), and the others are based on automatically tagging and parsing five million paraphrases from ParaNMT-50M; we describe these tasks in detail below.

For hyperparameters, the dimensions of the latent variables are 50. The dimensions of word embeddings are 50. We use cosine similarity as similarity metric for all of our experiments. We tune the weights for PRL and reconstruction loss from 0.1 to 1 in increments of 0.1 based on the development set performance. We use one sample from each latent variable during training. When evaluating VGVAE based models on STS tasks, we use the mean direction of the semantic variable y, while for syntactic similarity tasks, we use the mean vector of the syntactic variable z.

6.2 Baselines

Our baselines are a simple word averaging (WORDAVG) model and bidirectional LSTM averaging (BLSTMAVG) model, both of which have been shown to be very competitive for modeling semantic similarity when trained on paraphrases (Wieting and Gimpel, 2018). Specifically, WORDAVG takes the average over the word embeddings in the input sequence to obtain the sentence representation. BLSTMAVG uses the averaged hidden states of a bidirectional LSTM as the sentence representation, where forward and backward hidden states are concatenated. These models use 50 dimensional word embeddings and 50 dimensional LSTM hidden vectors per direction. These baselines are trained with DPL only. Additionally, we scramble the input sentence for BLSTMAVG since it has been reported beneficial for its performance in semantic similarity tasks (Wieting and Gimpel, 2017).

We also benchmark several pretrained embeddings on both semantic similarity and syntactic similarity datasets, including GloVe (Pennington et al., 2014),[3] SkipThought (Kiros et al., 2015),[4]

[3] We use 300 dimensional Common Crawl embeddings available at nlp.stanford.edu/projects/glove

[4] github.com/ryankiros/skip-thoughts

	semantic var.		syntactic var.	
	bm	avg	bm	avg
GloVe	39.0	48.7	-	-
SkipThought	42.1	42.0	-	-
InferSent	67.8	61.0	-	-
ELMo	57.7	60.3	-	-
BERT	4.5	15.0	-	-
WORDAVG	71.9	64.8	-	-
BLSTMAVG	71.4	64.4	-	-
VGVAE	45.5	42.7	40.8	43.2
VGVAE + WPL	51.5	49.3	28.1	31.0
VGVAE + DPL	68.4	58.2	37.8	40.5
VGVAE + PRL	67.9	57.8	29.6	32.7
VGVAE + PRL + WPL	69.8	61.3	23.2	27.9
VGVAE + PRL + DPL	71.2	64.2	31.7	33.9
VGVAE + DPL + WPL	71.0	63.5	24.1	29.0
ALL	72.3	65.1	20.1	24.2
ALL + LSTM enc.	72.5	65.1	16.3	24.5
ALL + LSTM enc. & dec.	**72.9**	**65.5**	**11.3**	**19.3**

Table 1: Pearson correlation (%) for STS test sets. bm: STS benchmark test set. avg: the average of Pearson correlation for each domain in the STS test sets from 2012 to 2016. Results are in bold if they are highest in the "semantic variable" columns or lowest in the "syntactic variable" columns. "ALL" indicates all of the multi-task losses are used.

InferSent (Conneau et al., 2017),[5] ELMo (Peters et al., 2018),[6] and BERT (Devlin et al., 2018).[7] For GloVe, we average word embeddings to form sentence embeddings. For ELMo, we average the hidden states from three layers and then average the hidden states across time steps. For BERT, we use the averaged hidden states from the last attention block.

7 Results and Analysis

7.1 Semantic Similarity

As shown in Table 1, the semantic and syntactic variables of our base VGVAE model show similar performance on the STS test sets. As we begin adding multi-task losses, however, the performance of these two variables gradually diverges, indicating that different information is being captured in the two variables. More interestingly, note that when *any* of the three losses is added to the base VGVAE model (even the WPL loss which makes no use of paraphrases), the performance of the semantic variable increases and the performance of the syntactic variable decreases; this suggests that each loss is useful in encouraging the latent variables to learn complementary information.

Indeed, the trend of additional losses both increasing semantic performance and decreasing syntactic performance holds even as we use more than two losses, except for the single case of VGVAE + PRL + DPL, where the syntactic performance increases slightly. Finally, we see that when the bag-of-words VGVAE model is used with all of the multi-task losses ("ALL"), we observe a large gap between the performance of the semantic and syntactic latent variables, as well as strong performance on the STS tasks that outperforms all baselines.

Using LSTM modules further strengthens the disentanglement between the two variables and leads to even better semantic performance. While using an LSTM encoder and a bag-of-words decoder is difficult to justify from a generative modeling perspective, we include results with this configuration to separate out the contributions of the LSTM encoder and decoder.

7.2 Syntactic Similarity

So far, we have only confirmed empirically that the syntactic variable has learned to *not* capture semantic information. To investigate what the syntactic variable has captured, we propose several syntactic similarity tasks.

In particular, we consider using the syntactic latent variable in calculating nearest neighbors for a 1-nearest-neighbor syntactic parser or part-of-speech tagger. We use our latent variables to define the similarity function in these settings and evaluate the quality of the output parses and tag sequences using several metrics.

Our first evaluation involves constituency parsing, and we use the standard training and test splits from the Penn Treebank. We predict a parse tree for each sentence in the test set by finding its nearest neighbor in the training set based on the cosine similarity of the mean vectors for the syntactic variables. The parse tree of the nearest neighbor will then be treated as our prediction for the test sentence. Since the train and test sentences may differ in length, standard parse evaluation metrics are not applicable, so we use tree edit distance (Zhang and Shasha, 1989)[8] to compute the distance between two parse tree without consider-

[5] We use model V1 available at github.com/facebookresearch/InferSent

[6] We use the original model available at allennlp.org/elmo

[7] We use bert-large-uncased available at github.com/huggingface/pytorch-pretrained-BERT

[8] github.com/timtadh/zhang-shasha

	Constituent Parsing (TED, ↓)		Constituent Parsing (F_1, ↑)		POS Tagging (%Acc., ↑)	
GloVe	120.8		27.3		23.9	
SkipThought	99.5		30.9		29.6	
InferSent	138.9		28.0		25.1	
ELMo	103.8		30.4		27.8	
BERT	101.7		28.6		25.4	
Random baseline	121.4		19.2		12.9	
Upper bound performance	51.6		71.1		62.3	
WORDAVG	107.0		25.5		21.4	
BLSTMAVG	106.8		25.7		21.6	
	semantic var.	syntactic var.	semantic var.	syntactic var.	semantic var.	syntactic var.
VGVAE	109.3	111.4	25.2	25.0	21.1	21.0
VGVAE + WPL	112.3	105.9	**24.1**	28.2	**20.3**	24.2
VGVAE + DPL	108.1	110.6	25.1	26.1	21.3	21.8
VGVAE + PRL	111.9	110.9	24.7	26.9	21.0	22.2
VGVAE + DPL + WPL	111.2	105.0	25.1	28.8	21.5	24.6
VGVAE + PRL + DPL	108.0	110.4	25.0	26.2	21.1	22.1
VGVAE + PRL + WPL	109.4	105.1	24.4	28.1	20.6	23.6
ALL	110.0	104.7	25.4	29.3	21.4	25.5
ALL + LSTM enc.	112.0	101.0	25.7	37.3	22.1	34.0
ALL + LSTM enc. & dec.	**114.6**	**100.5**	25.3	**38.8**	21.4	**35.7**

Table 2: Syntactic similarity evaluations, showing tree edit distance (TED) and labeled F_1 score for constituent parsing, and accuracy (%) for part-of-speech tagging. Numbers are bolded if they are worst in the "semantic variable" column or best in the "syntactic variable" column. "ALL" indicates all the multi-task losses are used.

ing word tokens.

To better understand the difficulty of this task, we introduce two baselines. The first randomly selects a training sentence. We calculate its performance by running it ten times and then reporting the average. We also report the upper bound performance given the training set. Since computing tree edit distance is time consuming, we subsample 100 test instances and compute the minimum tree edit distance for each sampled instance. Thus, this number can be seen as the approximated upper bound performance for this task given the training set.

To use a more standard metric for these syntactic similarity tasks, we must be able to retrieve training examples with the same number of words as the sentence we are trying to parse. We accordingly parse and tag the five million paraphrase subset of the ParaNMT training data using Stanford CoreNLP (Manning et al., 2014). To form a test set, we group sentences in terms of sentence length and subsample 300 sentences for each sentence length. After removing the paraphrases of the sentences in the test set, we use the rest of the training set as candidate sentences for nearest neighbor search, and we restrict nearest neighbors to have the same sentence length as the sentence we are attempting to parse or tag, which allows us to use standard metrics like labeled F_1 score and tagging accuracy for evaluation.

7.2.1 Results

As shown in Table 2, the syntactic variables and semantic variables demonstrate similar trends across these three syntactic tasks. Interestingly, both DPL and PRL help to improve the performance of the syntactic variables, even though these two losses are only imposed on the semantic variables. We saw an analogous pattern in Table 1, which again suggests that by pushing the semantic variables to learn information shared by paraphrastic sentences, we also encourage the syntactic variables to capture complementary syntactic information. We also find that adding WPL brings the largest improvement to the syntactic variable, and keeps the syntactic information carried by the semantic variables at a relatively low level. Finally, when adding all three losses, the syntactic variable shows the strongest performance across the three tasks.

In addition, we observe that the use of the LSTM encoder improves syntactic performance by a large margin and the LSTM decoder improves further, which suggests that the use of the LSTM decoder contributes to the amount of syntactic information represented in the syntactic variable.

Among pretrained representations, SkipThought shows the strongest performance overall and ELMo has the second best performance in the last two columns. While InferSent performs worst in the first column, it gives reasonable performance for the other two. BERT performs

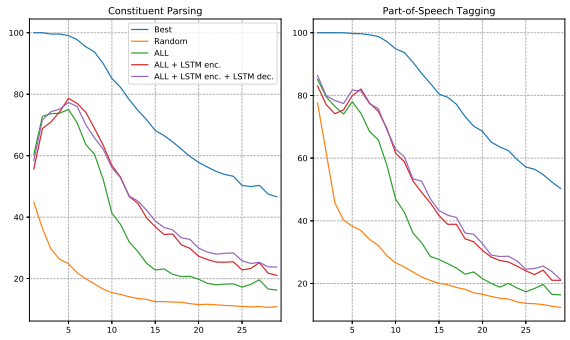

Figure 4: Constituency parsing F_1 scores and part-of-speech tagging accuracies by sentence length, for 1-nearest neighbor parsers based on semantic and syntactic variables, as well as a random baseline and an oracle nearest neighbor parser ("Best").

relatively well in the first column but worse in the other two.

To investigate the performance gap between the bag-of-words VGVAE and VGVAE with LSTM modules, in Figure 4 we plot the performance of our models and baselines as the length of the target sentence increases. We see that performance in all settings degrades as the sentences get longer. This may be due to the fact that the data is much sparser as sentence length increases (leaving fewer candidate nearest neighbors for prediction). We also see that above 4 words or so the performance gap between the bag-of-words VGVAE and VGVAE with LSTM modules becomes more and more obvious. This may be because the bag-of-words encoder has a harder time capturing syntactic information as sentence length increases. In addition, there is a slight improvement from using an LSTM decoder when the sentence length increases beyond 12 or so, which suggests that a bag-of-words decoder may struggle to capture certain parts of the syntactic information in the sentence, even when using an LSTM encoder.

7.3 Qualitative Analysis

To qualitatively evaluate our latent variables, we find (via cosine similarity) nearest neighbor sentences to test set examples in terms of both the semantic and syntactic representations. We also find nearest neighbors of words (which we view as single-word sentences). We discuss the results of this analysis below.

7.3.1 Lexical Analysis

Table 3 shows word nearest neighbors for both syntactic and semantic representations. We see that the most similar words found by the syntactic variable share the same part-of-speech tags with the query words. For example, "starting" is close to "getting" and "taking," even though these words are not semantically similar. Words retrieved according to the semantic variable, however, are more similar semantically, e.g., "begin" and "starts". As another example, "times" is similar to words that are either related to descriptions of frequency (e.g., "twice" and "often") or related to numbers (e.g., "thousand", "seven").

7.3.2 Sentential Analysis

As shown in Table 4, sentences that are similar in terms of their semantic variables tend to have similar semantics. However, sentences that are similar in terms of their syntactic variables are mostly semantically unrelated but have similar surface forms. For example, "you 're gon na save her life ." has the same meaning as "you will save her ." while having a similar syntactic structure to "you 're gon na give a speech ." (despite having very different meanings). As another example, although the semantic variable does not find a good match for "i have much more colours at home .", which can be attributed to the limited size of candidate sentences, the nearest syntactic neighbor ("you have a beautiful view from here .") has a very similar syntactic structure to the query sentence.

8 Discussion

In this paper we explored simple methods to disentangle syntax and semantics in latent representations of sentences. One goal was to measure the impact of simple decisions on the disentanglement of both the semantic and syntactic variables, even when restricting ourselves to simplified bag-of-words encoders. Due to the constrained nature of these bag-of-words models, we found that it was important to use different word embedding spaces for the semantic and syntactic encoders. In preliminary experiments, we experimented with the use of the same word embedding space but distinct feed-forward layers in the two latent variable encoders. However, this setting proved extremely difficult to achieve a disentanglement between syntax and semantics. Hence an important component of disentanglement with these bag-of-words encoders is the use of different word embedding spaces.

starting	*syntactic:* getting heading sitting chasing taking require trying sharing bothering pushing paying *semantic:* begin start stopping forward rising wake initial starts goes started again getting beginning
area	*syntactic:* engines certificate guests bottle responsibility lesson pieces suit bags vessel applications *semantic:* sector location zone fields rooms field places yard warehouse seats coordinates territory
considered	*syntactic:* stable limited odd scary classified concerned awful purple impressive embarrassing jealous *semantic:* thought assumed regard reasons wished understood purposes seemed expect guessed meant
jokes	*syntactic:* gentlemen photos finding baby missile dna parent shop murder science recognition sheriff *semantic:* funny humor prize stars cookie paradise dessert worthy smile happiness thrilled ideal kidding
times	*syntactic:* princess officer wounds plan gang ships feelings user liar elements coincidence degrees pattern *semantic:* twice later thousand pages seven every once often decade forgotten series four eight day time

Table 3: Examples of the most similar words to particular query words using syntactic variable (first row) or semantic variable (second row).

Query Sentence	Semantically Similar	Syntactically Similar
i have much more colours at home .	even if there was food , would n't it be at least 300 years old ?	you have a beautiful view from here .
victor had never known darkness like it .	he had never experienced such darkness as this .	you seem like a really nice kid .
this is , uh , too serious .	but this is too serious .	it is , however , illegal discrimination .
you 're gon na save her life .	you will save her .	you 're gon na give a speech .
we 've got to get a move on .	come on , we got ta move .	you 'll have to get in there .
and that was usually the highlight of my day .	i really enjoyed it when i did it .	and yet that was not the strangest aspect of the painting .
we do need to collect our taxes somehow .	we have to earn the money we need .	now i have to do my job .
this is just such a surprise .	oh . this is a surprise .	this is just a little gain .
okay . aw , that 's so romantic .	it 's so romantic !	oh . well , that 's not good .
we 're gon na have to do something about this .	we 'll have to do something about that .	we 're gon na have to do something about yours .

Table 4: Examples of most similar sentences to particular query sentences in terms of the semantic variable or the syntactic variable.

We also conducted experiments using LSTM encoders and decoders as recurrent neural networks are a natural way to capture syntactic information in a sentence. We found this approach to give us additional benefits for both disentangling semantics and syntax and achieving better results overall. Nonetheless, we find it encouraging that even when using bag-of-words encoders, our multi-task losses are able to achieve a separation as measured by our semantic and syntactic similarity tasks.

9 Conclusion

We proposed a generative model and several losses for disentangling syntax and semantics in sentence representations. We also proposed syntactic similarity tasks for measuring the amount of disentanglement between semantic and syntactic representations. We characterized the effects of the losses as well as the use of LSTM modules on both semantic tasks and syntactic tasks. Our models achieve the best performance across both sets of similarity tasks when the latent representations are most disentangled.

Acknowledgments

We would like to thank the anonymous reviewers, NVIDIA for donating GPUs used in this research, and Google for a faculty research award to K. Gimpel that partially supported this research.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Inigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 2015. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 252–263. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 81–91. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab,

Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43. Association for Computational Linguistics.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*.

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.

Marcel Bollmann, Anders Søgaard, and Joachim Bingel. 2018. Multi-task learning for historical text normalization: Size matters. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 19–24. 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.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Karen Livescu, and Kevin Gimpel. 2018. Variational sequential labelers for semi-supervised learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 215–226. Association for Computational Linguistics.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 2180–2188, USA. Curran Associates Inc.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised pos induction: How far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 575–584. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680. Association for Computational Linguistics.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. 2018. Hyperspherical variational auto-encoders. *34th Conference on Uncertainty in Artificial Intelligence (UAI-18)*.

Michel Deudon. 2018. Learning semantic similarity in a continuous space. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 993–1004. Curran Associates, Inc.

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

Jiachen Du, Wenjie Li, Yulan He, Ruifeng Xu, Lidong Bing, and Xuan Wang. 2018. Variational autoregressive decoder for neural response generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3154–3163.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *AAAI*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Proceedings of NIPS*.

Anirudh Goyal Alias Parth Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. 2017. Z-forcing: Training stochastic recurrent networks. In *Advances in neural information processing systems*, pages 6713–6723.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, and Aaron Courville Vincent Dumoulin. 2017. Improved Training of Wasserstein GANs. In *Proceedings of NIPS*.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Wei-Ning Hsu, Yu Zhang, and James Glass. 2017. Unsupervised learning of disentangled and interpretable representations from sequential data. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1878–1889. Curran Associates, Inc.

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.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. 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 1875–1885. Association for Computational Linguistics.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2018. Disentangled representation learning for text style transfer. *arXiv preprint arXiv:1808.04339*.

Daniel Jurafsky. 1988. Issues in relating syntax and semantics. In *Coling Budapest 1988 Volume 1: International Conference on Computational Linguistics*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial Autoencoders. *arXiv:1511.05644*.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330.

Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. 2016. Disentangling factors of variation in deep representation using adversarial training. In *Advances in Neural Information Processing Systems*, pages 5040–5048.

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.

Jeff Mitchell. 2016. Decomposing bilexical dependencies into semantic and syntactic vectors. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 127–136. Association for Computational Linguistics.

Jeff Mitchell and Mark Steedman. 2015. Orthogonality of syntax and semantics within distributional spaces. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1301–1310. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237. 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.

Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. 2014. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning*, pages 1431–1439.

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 J. Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of ICML*.

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.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, pages 6830–6841.

Joshua B Tenenbaum and William T Freeman. 2000. Separating style and content with bilinear models. *Neural computation*, 12(6):1247–1283.

Robert D. van Valin, Jr. 2005. *Exploring the Syntax-Semantics Interface*. Cambridge University Press.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *Proceedings of International Conference on Learning Representations*.

John Wieting and Kevin Gimpel. 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2078–2088, Vancouver, Canada. Association for Computational Linguistics.

John Wieting and Kevin Gimpel. 2018. ParaNMT-50M: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462. Association for Computational Linguistics.

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

Mike Wu and Noah Goodman. 2018. Multimodal generative models for scalable weakly-supervised learning. In *Advances in Neural Information Processing Systems*.

Jiacheng Xu and Greg Durrett. 2018. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4503–4513. Association for Computational Linguistics.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. Structvae: Tree-structured latent variable models for semi-supervised semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 754–765. Association for Computational Linguistics.

Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing*, 18(6):1245–1262.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander M Rush, and Yann LeCun. 2018. Adversarially Regularized Autoencoders. In *Proceedings of ICML*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 654–664.

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.