SNLI/PI

A Cross-Sentence Latent Variable Model for Semi-Supervised Text Sequence Matching

Jihun Choi, Taeuk Kim, Sang-goo Lee
Department of Computer Science and Engineering
Seoul National University
{jhchoi,taeuk,sglee}@europa.snu.ac.kr

Abstract

We present a latent variable model for predicting the relationship between a pair of text sequences. Unlike previous auto-encoding–based approaches that consider each sequence separately, our proposed framework utilizes both sequences within a single model by generating a sequence that has a given relationship with a source sequence. We further extend the cross-sentence generating framework to facilitate semi-supervised training. We also define novel semantic constraints that lead the decoder network to generate semantically plausible and diverse sequences. We demonstrate the effectiveness of the proposed model from quantitative and qualitative experiments, while achieving state-of-the-art results on semi-supervised natural language inference and paraphrase identification.

提出了隐变量模型来预侧两个句子之间的关系。不像之前基于自编码方法，分别考虑两个文本序列。本文提出的架构是在单一模型内利用两个序列，生成一个和原序列有给定关系的序列。

1 Introduction

Text sequence matching is a task whose objective is to predict the degree of match between two or more text sequences. For example, in natural language inference, a system has to infer the relationship between a premise and a hypothesis sentence, and in paraphrase identification a system should find out whether a sentence is a paraphrase of the other. Since various natural language processing problems, including answer sentence selection, text retrieval, and machine comprehension, involve text sequence matching components, building a high-performance text matching model plays a key role in enhancing quality of systems for these problems (Tan et al., 2016; Rajpurkar et al., 2016; Wang and Jiang, 2017; Tymoshenko and Moschitti, 2018).

With the emergence of large-scale corpora, end-to-end deep learning models are achieving remarkable results on text sequence matching; these include architectures that are linguistically motivated (Bowman et al., 2016a; Chen et al., 2017a; Kim et al., 2019), that introduce external knowledge (Chen et al., 2018), and that use attention mechanisms (Parikh et al., 2016; Shen et al., 2018b). The recent deep neural network–based work on text matching could roughly be categorized into two subclasses: i) methods that exploit inter-sentence features and ii) methods based on sentence encoders. In this work, we focus on the latter where sentences[1] are separately encoded using a shared encoder and then fed to a classifier network, due to its efficiency and general applicability across tasks.

本文是使用一个共享的编码器分别编码两个句子，再使用分类器。

最近的基于深度神经网络的文本匹配模型可分为两类：
①.利用句子间的特征的方法。
②.基于句子编码的方法。

Meanwhile, despite the success of deep neural networks in natural language processing, the fact that they require abundant training data might be problematic, as constructing labeled data is a time-consuming and labor-intensive process. To mitigate the data scarcity problem, several semi-supervised learning paradigms, that take advantage of unlabeled data when only some of the data examples are labeled (Chapelle et al., 2010), are proposed. These unlabeled data are much easier to collect, thus utilizing them could be a good option; for example in text matching, possibly related sentence pairs could be retrieved from a database of text via simple heuristics such as word overlap.

In this paper, we propose a cross-sentence latent variable model for semi-supervised text sequence matching. The proposed framework is based on deep probabilistic generative models (Kingma and Welling, 2014; Rezende et al., 2014) and is extended to make use of unlabeled data. As it is trained to generate a sentence that has a given relationship with a source sentence, both sentences in a pair are utilized together, and thus training objectives are defined more naturally than other models that consider each sentence separately (Zhao et al., 2018; Shen et al., 2018a). To further regularize

训练目标

[1] Throughout the paper, we will use the term ‘sequence’ and ‘sentence’ interchangeably unless ambiguous.

语义约束

the model to generate more plausible and diverse sentences, we define semantic constraints and use them for fine-tuning.

From experiments, we empirically prove that the proposed method significantly outperforms previous work on semi-supervised text sequence matching. We also conduct extensive qualitative analyses to validate the effectiveness of the proposed model.

The rest of the paper is organized as follows. In §2, we briefly introduce the background for our work. We describe the proposed cross-sentence latent variable model in §3, and give results from experiments in §4. We study the prior work related to ours in §5 and conclude in §6.

2 Background

2.1 Variational Auto-Encoders

Variational auto-encoder (VAE, Kingma and Welling, 2014) is a deep generative model for modeling the data distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$. It assumes that a data point $\mathbf{x}$ is generated by the following random process: (1) $\mathbf{z}$ is sampled from $p(\mathbf{z})$ and (2) $\mathbf{x}$ is generated from $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$.

Thus the natural training objective would be to directly maximize the marginal log-likelihood $\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \int_{\mathbf{z}} p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$. However it is intractable to compute the marginal log-likelihood without using simplifying assumption such as mean-field approximation (Blei et al., 2017). Therefore the following variational lower bound $-\mathcal{L}$ is used as a surrogate objective:

$$\begin{aligned} -\mathcal{L}(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x}) = &-D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})) \\ &+ \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right], \end{aligned}$$

where $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ is a variational approximation to the unknown $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, and $D_{KL}(q\|p)$ is the Kullback-Leibler (KL) divergence between q and p. Maximizing the surrogate objective $-\mathcal{L}$ is proven to minimize $D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))$, and it can also be seen as maximizing the expected data log-likelihood with respect to $q_{\boldsymbol{\phi}}$ while using $D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}))$ as a regularization term.

VAEs are successfully applied in modeling various data: including image (Pu et al., 2016; Gulrajani et al., 2017), music (Roberts et al., 2018), and text (Miao et al., 2016; Bowman et al., 2016b). The VAE framework can also be extended to constructing conditional generative models (Sohn et al., 2015) or learning from semi-supervised data (Kingma et al., 2014; Xu et al., 2017).

VAEs for text pair modeling. The most simple approach to modeling text pairs using the VAE framework is to consider two text sequences separately (Zhao et al., 2018; Shen et al., 2018a). That is, a generator is trained to reconstruct a single input sequence rather than integrating both sequences, and the two latent representations encoded from a variational posterior are given to a classifier network. When label information is not available, only the reconstruction objective is used for training. This means that the classifier parameters are not updated in the unsupervised setting, and thus the interaction between the variational posterior (or encoder) and the classifier could be restricted.

2.2 von Mises–Fisher Distribution

Since the advent of deep generative models with variational inference, the typical choice for prior and variational posterior distribution has been the Gaussian, likely due to its well-studied properties and easiness of reparameterization. However it often leads a model to face the posterior collapse problem where a model ignores latent variables by pushing the KL divergence term to zero (Chen et al., 2017b; van den Oord et al., 2017), especially in text generation models where powerful decoders are used (Bowman et al., 2016b; Yang et al., 2017).

Various techniques are proposed to mitigate this problem: including KL cost annealing (Bowman et al., 2016b), weakening decoders (Yang et al., 2017), skip connection (Dieng et al., 2019), using different objectives (Alemi et al., 2018), and using alternative distributions (Guu et al., 2018). In this work, we take the last approach by utilizing a von Mises–Fisher (vMF) distribution.

A vMF distribution is a probability distribution on the $(d-1)$-sphere, therefore samples are compared according to their directions, reminiscent of the cosine similarity. It has two parameters—mean direction $\boldsymbol{\mu} \in \mathbb{R}^d$ and concentration $\kappa \in \mathbb{R}$. As the KL divergence between $\text{vMF}(\boldsymbol{\mu}, \kappa)$ and the hyperspherical uniform distribution $\mathcal{U}(S^{d-1}) = \text{vMF}(\cdot, 0)$ only depends on κ, the KL divergence is a constant if the concentration parameter is fixed. Therefore when $\text{vMF}(\boldsymbol{\mu}, \kappa)$ with fixed κ and $\text{vMF}(\cdot, 0)$ are used as posterior and prior, the posterior collapse does not occur inherently.

To the best of our knowledge, Guu et al. (2018) were the first to use vMF as posterior and prior

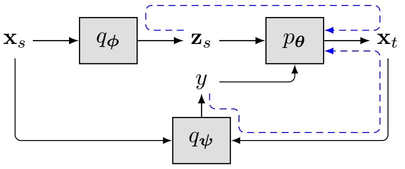
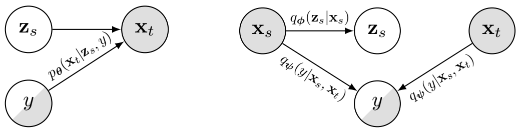


Figure 1: The overview of the entire framework. Blue dashed lines indicate semantic constraints.


$\mathbf{z}_s$
$\mathbf{x}_t$
y
$p_\theta(\mathbf{x}_t|\mathbf{z}_s, y)$
$\mathbf{x}_s$
$q_\phi(\mathbf{z}_s|\mathbf{x}_s)$
$\mathbf{z}_s$
$\mathbf{x}_t$
$q_\psi(y|\mathbf{x}_s, \mathbf{x}_t)$
y
$q_\psi(y|\mathbf{x}_s, \mathbf{x}_t)$

(a) Generation model (b) Recognition model

Figure 2: Illustration of the graphical models. (a) the generative process of the output $\mathbf{x}_t$; (b) the approximate inference of $\mathbf{z}_s$ and the discriminative classifier for y.

for VAEs, and Xu and Durrett (2018) empirically proved the effectiveness of vMF-VAE in natural language generation. Davidson et al. (2018) generalized the vMF-VAE and proposed the reparameterization trick for vMF. We refer readers to Appendix A for detailed description of vMF we used.

3 Proposed Framework

In this section, we describe the proposed framework in detail. We formally define the cross-sentence latent variable model (CS-LVM) and describe the optimization objectives. We also introduce semantic constraints to keep learned representations in a semantically plausible region. Fig. 1 illustrates the entire framework.

3.1 Cross-Sentence Latent Variable Model

Though the auto-encoding frameworks described in §2.1 have intriguing properties, it may hinder the possibility of training an encoder to extract rich features for text pair modeling, due to the fact that the generative modeling process is confined within a single sequence. Therefore the interaction between a generative model and a discriminative classifier is restricted, since the two sequences are separately modeled and the pair-wise information is only considered through the classifier network.

Our proposed CS-LVM addresses this problem by cross-sentence generation of text given a text pair and its label. As the sentences in a pair are directly related within a generative model, the training objectives are defined in a more principled way than VAE-based semi-supervised text matching frameworks. Notably it also mimics the dataset construction process of some corpora: *a worker generates a target text given a label and a source text* (e.g. Bowman et al., 2015; Williams et al., 2018).

Given a pair $(\mathbf{x}_1, \mathbf{x}_2)$, let $\mathbf{x}_s, \mathbf{x}_t \in \{\mathbf{x}_1, \mathbf{x}_2\}$ be a source and a target sequence respectively. Then we assume $\mathbf{x}_t$ is generated according to the following process (see Fig. 2a):

1. a latent variable $\mathbf{z}_s$ that contains the content of a source sequence is sampled from $p(\mathbf{z}_s)$,
2. a variable y that determines the relationship between a target and the source sequence is sampled from $p(y)$,
3. $\mathbf{x}_t$ is generated from a conditional distribution $p_\theta(\mathbf{x}_t|\mathbf{z}_s, y)$.

Zs包含原序列的内容

y决定原序列和目标序列之间的关系

Xt是生成的目标序列.

In the above process, the class label y is treated as a hidden variable in the unsupervised case and an observed variable in the supervised case.

Accordingly, when the label information is available, the optimization objective for a generative model is the marginal log-likelihood of the observed variables $\mathbf{x}_t$ and y:

监督情况下，即y是可见的情况下，目标函数

$$\begin{aligned}\log p_\theta(\mathbf{x}_t, y) &= \log \int p_\theta(\mathbf{x}_t, \mathbf{z}_s, y) d\mathbf{z}_s \\ &= \log \int p_\theta(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y) d\mathbf{z}_s. \quad (1)\end{aligned}$$

To address the intractability we instead optimize the lower bound of Eq. 1:[2]

难解性

$$\begin{aligned}\log p_\theta(\mathbf{x}_t, y) \geq &-D_{KL}(q_\phi(\mathbf{z}_s|\mathbf{x}_s) \| p(\mathbf{z}_s)) \\ &+ \mathbb{E}_{q_\phi(\mathbf{z}_s|\mathbf{x}_s)}[\log p_\theta(\mathbf{x}_t|y, \mathbf{z}_s)] + \log p(y), \quad (2)\end{aligned}$$

where $q_\phi(\mathbf{z}_s|\mathbf{x}_s)$ is a variational approximation of the posterior $p_\theta(\mathbf{z}_s|\mathbf{x}_t, y)$. Though Eq. 2 holds for any q_ϕ having the same support with $p(\mathbf{z}_s)$, we choose this form of variational posterior from the following motivation: *since* $\mathbf{x}_s$ *is related to* $\mathbf{x}_t$ *by the label information* y, $\mathbf{x}_s$ *would have an influence on the space of* $\mathbf{z}_s$ *in a similar way to* $(\mathbf{x}_t, y)$. Due to this particular choice of q_ϕ, $\mathbf{z}_s$ depends only on $\mathbf{x}_s$ and is independent of the label information possibly permeated in $\mathbf{x}_t$. In other words, this design induces q_ϕ to extract the features needed for controlling the semantics only from $\mathbf{x}_s$, while preventing q_ϕ from encoding other biases.

变分近似7w

通过这样特殊的选择qφ，Zs仅依赖于Xs，和y无关.

To extend the objective to the unsupervised setup, we marginalize out y from Eq. 2 using a

[2] See Appendix B for derivation of the lower bound.

classifier distribution. We will provide more detailed explanation of the optimization objectives in §3.3.

3.2 Architecture

Now we describe the architectures we used for constructing CS-LVM. We first encode a source sequence into a fixed-length representation using a recurrent neural network (RNN): $g^{enc}(\mathbf{x}_s) = \mathbf{m}_s$. From $\mathbf{m}_s$ we obtain a variational approximate distribution $q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s) = g^{code}(\mathbf{m}_s)$ and sample a latent representation $\mathbf{z}_s \sim q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)$. In our experiments, a long short-term memory (LSTM) recurrent network and a feed-forward network are used as g^{enc} and g^{code} respectively. From the fact that the mean direction parameter $\boldsymbol{\mu}_s$ of vMF$(\boldsymbol{\mu}_s, \kappa)$ should be a unit vector, g^{code} additionally normalizes the output of the feed-forward network to be $\|g^{code}(\mathbf{m}_s)\|_2 = 1$. ???

$$\begin{tikzcd} X_s \arrow[d, "g^{enc}()"] \\ m_s \arrow[d, "g^{code}()"] \\ \text{标准化 } \|\cdot\|_2=1 \arrow[d] \\ V_i \arrow[d, "g^{dec}()"] \\ S_i \arrow[d, "g^{out}()"] \\ X_t \end{tikzcd}$$

$W_{t,i}$ → V_i ← y

Then we generate the target sequence $\mathbf{x}_t$ from $\mathbf{z}_s$ and y. Similarly to the encoder network, we use an LSTM for a decoder, thus the distribution is factorized as follows:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_t|y, \mathbf{z}_s) = \prod_{i=1}^{N_{\mathbf{x}_t}+1} p_{\boldsymbol{\theta}}(w_{t,i}|w_{t,<i}, y, \mathbf{z}_s), \quad (3)$$

where $\mathbf{x}_t = (x_{t,1}, \ldots, x_{t,N_{\mathbf{x}_t}})$ and $w_{t,0} =$ <s>, $w_{t,N_{\mathbf{x}_t}+1} =$ </s> are special tokens indicating the start and the end of a sequence.

We project the word index $w_{t,i}$ and label index y into embedding spaces to obtain the word embedding $\mathbf{w}_{t,i}$ and label embedding $\mathbf{y}$. Then to construct an input for i-th time step, $\mathbf{v}_t$, we concatenate the i-th target word embedding $\mathbf{w}_{t,i}$, the label embedding $\mathbf{y}$, and the latent representation $\mathbf{z}_s$ altogether:

$$\mathbf{v}_i = [\mathbf{w}_{t,i}; \mathbf{y}; \mathbf{z}_s].$$

Thus $p_{\boldsymbol{\theta}}(w_{t,i}|w_{t,<i}, \mathbf{z}_s, y)$ is computed from i-th state $\mathbf{s}_i$ of the decoder RNN:

$$p_{\boldsymbol{\theta}}(w_{t,i}|w_{t,<i}, y, \mathbf{z}_s) = \mathrm{softmax}(g^{out}(\mathbf{s}_i))$$

$$\mathbf{s}_i = g_i^{dec}(\mathbf{v}_i, \mathbf{s}_{i-1}),$$

where g^{out} is a feed-forward network and g_i^{dec} is the state transition function of the decoder LSTM at i-th time step.

For a discriminative classifier network we follow the siamese architecture, as mentioned in §1. $\mathbf{x}_s$ and $\mathbf{x}_t$ are fed to a shared LSTM network f^{enc} to obtain sentence vectors $\mathbf{h}_1 = f^{enc}(\mathbf{x}_s)$ and $\mathbf{h}_2 = f^{enc}(\mathbf{x}_t)$. Then $\mathbf{h}_1$ and $\mathbf{h}_2$ are combined by the function f^{fuse} to form a single fused vector, and the fused representation is given to a feed-forward network f^{disc} to infer the relationship:

Algorithm 1 Training procedure of CS-LVM.

```
Input: Labeled dataset $\mathcal{X}_l$, Unlabeled dataset $\mathcal{X}_u$,
       Model parameters $\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}$
 1: procedure TRAIN($\mathcal{X}_l, \mathcal{X}_u, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}$)
 2:     repeat
 3:         Sample $(\mathbf{x}_{l,s}, \mathbf{x}_{l,t}, y_l) \sim \mathcal{X}_l$
 4:         Sample $(\mathbf{x}_{u,s}, \mathbf{x}_{u,t}) \sim \mathcal{X}_u$
 5:         Compute $\mathcal{L}_l(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_{l,s}, \mathbf{x}_{l,t}, y_l)$ by (6)
 6:         Compute $\mathcal{L}_u(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_{u,s}, \mathbf{x}_{u,t})$ by (9)
 7:         Update $\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}$ by gradient descent on $\mathcal{L}_l + \mathcal{L}_u$
 8:     until stop criterion is met
 9: procedure FINETUNE($\mathcal{X}_l, \mathcal{X}_u, \boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}$)
10:     repeat
11:         Update $\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}$ following line 3–7
12:         Update $\boldsymbol{\theta}$ by gradient descent on (11–14)
13:     until stop criterion is met
```

$$q_{\boldsymbol{\psi}}(y|\mathbf{x}_1, \mathbf{x}_2) = \mathrm{softmax}(\underset{\text{FFN}}{\boxed{f^{disc}}}(\boxed{f^{fuse}}(\mathbf{h}_1, \mathbf{h}_2))).$$

把2个向量融合成1个向量.

To learn from data more efficiently and to reduce the number of trainable parameters, we tie the weights for two encoders—for the generative model and the discriminative classifier; i.e. $g^{enc} = f^{enc}$. This mitigates the problem that only source sequences are used for training g^{enc} and enhances the interaction between the generative model and the classifier. We will see from experiments that tying encoder weights improves performance and stabilizes optimization (§4.3).

Also note that the functions $g^{\square}$ are only used in training, and the model has the same test-time computational complexity with typical classification models.

generative model 只在训练时使用.
测试时只用 classification model部分.

3.3 Optimization

In this subsection we describe how the entire model is optimized. We first define optimization objectives for supervised and unsupervised training, and then introduce constraints to regularize the model to generate sequences with intended semantic characteristics. The entire optimization procedure is summarized in Algorithm 1.

3.3.1 Supervised Objective

In the supervised setting, a data sample is assumed to contain label information: $(\mathbf{x}_1, \mathbf{x}_2, y) \in \mathcal{X}_l$. Without loss of generality let us assume $(\mathbf{x}_s, \mathbf{x}_t) = (\mathbf{x}_1, \mathbf{x}_2)$.[3] Since y is an observed vari-

$$\begin{tikzcd} X_s \arrow[r, "f^{enc}()"] & h_1 \arrow[r, "f^{fuse}"] & {} \arrow[r, "f^{disc}"] & {} \arrow[r, "\text{Softmax}"] & \text{关系预测值} \\ X_t \arrow[r] & h_2 \end{tikzcd}$$

[3] The relationship between a source and a target may either be unidirectional, bidirectional, or reflexive, depending

able in this case, we can directly use Eq. 2 in training. From Eqs. 2 and 3, the objective for the generative model is defined by:[4]

generative model 的目标函数：

$$-\mathcal{L}_l^{gen}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}_s, \mathbf{x}_t, y) = \log p_{\boldsymbol{\theta}}(\mathbf{x}_t|y, \mathbf{z}_s) + \log p(y) - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)\|p(\mathbf{z}_s)), \quad (4)$$

where $\mathbf{z}_s \sim q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)$ and $p(y)$, $p(\mathbf{z}_s)$ are prior distributions of y, $\mathbf{z}_s$. Considering that we assume $p(y)$ to be a fixed uniform distribution of labels, the $\log p(y)$ term can be ignored in training: $\|\nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \log p(y)\|_2 = 0$.

均匀分布

For training, the typical teacher forcing method is used; i.e. ground-truth words are used as input words. We use vMF$(g^{code}(\mathbf{m}_s), \kappa)$ (κ: hyperparameter) for the variational posterior $q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)$ and vMF$(\cdot, 0)$ for the prior $p(\mathbf{z}_s)$.

The discriminator objective is defined as a conventional maximum likelihood:

classification model 的目标函数：

$$-\mathcal{L}_l^{disc}(\boldsymbol{\psi}; \mathbf{x}_s, \mathbf{x}_t, y) = \log q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t). \quad (5)$$

Finally, the two objectives are combined to construct the objective for supervised training:

监督学习的总目标函数：

$$\mathcal{L}_l(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_s, \mathbf{x}_t, y) = \mathcal{L}_l^{gen} + \lambda \mathcal{L}_l^{disc}, \quad (6)$$

where λ is a hyperparameter.

3.3.2 Unsupervised Objective

In this case, the model does not have an access to label information; a data point is represented by $(\mathbf{x}_s, \mathbf{x}_t) \in \mathcal{X}_u$ and thus y is a hidden variable. To facilitate the unsupervised training, we marginalize y out as below and derive the lower bound:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}_t) = \log \sum_y \int p_{\boldsymbol{\theta}}(\mathbf{x}_t, \mathbf{z}_s, y) d\mathbf{z}_s \geq \mathbb{E}_{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y,\mathbf{z}_s|\mathbf{x}_s,\mathbf{x}_t)} \left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t, \mathbf{z}_s, y)}{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \right]. \quad (7)$$

And from the assumption presented in the graphical model (Fig. 2b),

$$q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t) = q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s) q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t). \quad (8)$$

Finally we obtain the following lower bound for $\log p_{\boldsymbol{\theta}}(\mathbf{x}_t)$ from Eqs. 7 and 8:[5]

$$\mathcal{L}_u(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_s, \mathbf{x}_t) = -\mathcal{H}(q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t)) + \mathbb{E}_{q_{\boldsymbol{\psi}}(y|\mathbf{x}_s,\mathbf{x}_t)} \left[\mathcal{L}_l^{gen}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}_s, \mathbf{x}_t, y) \right]. \quad (9)$$

Here the second expectation term can be computed either by enumeration or sampling, and we used the former as the datasets we used have relatively small label sets (2 or 3) and it is known to yield better results than sampling (Xu et al., 2017). We will compare the two methods in §4.3.

To sum up, at every training iteration, given a labeled and unlabeled data sample $(\mathbf{x}_{l,s}, \mathbf{x}_{l,t}, y_l)$, $(\mathbf{x}_{u,s}, \mathbf{x}_{u,t})$, we optimize the following objective.

$$\mathcal{L} = \mathcal{L}_l(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_{l,s}, \mathbf{x}_{l,t}, y_l) + \mathcal{L}_u(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_{u,s}, \mathbf{x}_{u,t}) \quad (10)$$

3.3.3 Fine-Tuning with Semantic Constraints

Since the generator is trained via maximum likelihood training which considers all words in a sentence equivalently, the label information may not be reflected enough in generation owing to high-frequency words. For example in natural language inference, the word occurrences of the following three hypothesis sentences highly overlap, but they should have different relation with the premise.[6]

P: *A man is cutting metal with a tool .*
H1: *A man is cutting metal .*
H2: *A man is cutting metal with the wrong tool .*
H3: *A man is cutting metal with his mind .*

Thus for some data points, the strategy that only predicts words that overlap across hypotheses could receive a fairly high score, which might weaken the integration of y into the generator. To mitigate this, we fine-tune the trained generator using the following semantic constraint:

$$-\mathcal{R}^y(\boldsymbol{\theta}; \mathbf{x}_s, \mathbf{x}_t) = \log q_{\boldsymbol{\psi}}(\tilde{y}|\mathbf{x}_s, \widetilde{\mathbf{x}}_t), \quad (11)$$

where $\tilde{y} \sim p(y)$, $\mathbf{z}_s \sim q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)$, and $\widetilde{\mathbf{x}}_t = \operatorname{argmax}_{\mathbf{x}_t} p_{\boldsymbol{\theta}}(\mathbf{x}_t|\tilde{y}, \mathbf{z}_s)$. This constraint enforces the sequence $\widetilde{\mathbf{x}}_t$ generated by conditioning on $\tilde{y}$ and $\mathbf{z}_s$ to actually have the relationship $\tilde{y}$ with $\mathbf{x}_s$.

让生成的 $\tilde{X}_t$ 与 X_s 有关系 $\tilde{y}$.

We also introduce a constraint on $\mathbf{z}$ that keeps the distributions of $\widetilde{\mathbf{z}}_t$ (the latent content variable

on the characteristics of a task. For some experiments we additionally used swapped data examples, $(\mathbf{x}_s, \mathbf{x}_t) = (\mathbf{x}_2, \mathbf{x}_1)$, for training. We explain more on this in §4.

[4] Note that we define all objectives $\mathcal{L}$, $\mathcal{R}$ as *minimization objectives* to avoid confusion.

[5] See Appendix B for details.

[6] Examples are taken from the SNLI development set, pair ID `4904199439.jpg#{2r1e,2r1n,2r1c}`.

obtained by encoding the generated sequence $\widetilde{\mathbf{x}}_t$) and $\mathbf{z}_s$ close:

避免生成器
的 collapse 问题.

$$-\mathcal{R}^{\mathbf{z}}(\boldsymbol{\theta}; \mathbf{x}_s, \mathbf{x}_t) = \log q_{\boldsymbol{\phi}}(\mathbf{z}_t = \widetilde{\mathbf{z}}_t | \mathbf{x}_t), \qquad (12)$$

where $\widetilde{\mathbf{z}}_t \sim q_{\boldsymbol{\phi}}(\widetilde{\mathbf{z}}_t | \widetilde{\mathbf{x}}_t)$. In other words, it pushes the generated sequence $\widetilde{\mathbf{x}}_t$ to be in a similar semantic space with the ground-truth target sequence $\mathbf{x}_t$. Consequently, it can help alleviate the generator collapse problem where a generator produces only a handful of simple neutral patterns independent of the input sequence, by relating $\widetilde{\mathbf{z}}_t$ to $\mathbf{z}_t$.[7]

From similar motivation, we also add an additional constraint that encourages the generated sentences originating from different source sentences to be dissimilar. To reflect this, we define the following minibatch-level constraint that penalizes the mean direction vectors encoded from the generated sentences for being too close:

$$-\mathcal{R}^{\boldsymbol{\mu}}(\boldsymbol{\theta}; \mathcal{B}) = \mathbb{E}_{\mathcal{B}}[d({\boldsymbol{\mu}_t}^{(i)}, \bar{\boldsymbol{\mu}}_t)], \qquad (13)$$

where we denote values related to i-th sample of a minibatch $\mathcal{B}$ using superscript: $\square^{(i)}$. In the above, $\boldsymbol{\mu}_t^{(i)} = g^{code}(g^{enc}(\widetilde{\mathbf{x}}_t^{(i)}))$, $\bar{\boldsymbol{\mu}}_t = \sum_{i=1}^{|\mathcal{B}|} \boldsymbol{\mu}_t^{(i)} / |\mathcal{B}|$, and $d(\cdot, \cdot)$ is a distance measure between vectors. The mean direction vector $\boldsymbol{\mu}$ of vMF$(\boldsymbol{\mu}, \kappa)$ is on a unit hypersphere, so we use the cosine distance: $d(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) = 1 - \langle \boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \rangle$.

As the sequence generation process is not differentiable, the gradients from the semantic constraints cannot propagate to the generator parameters. To relax the discreteness, we use the Gumbel-Softmax reparameterization (Jang et al., 2017; Maddison et al., 2017). Using the Gumbel-Softmax trick, we obtain a continuous probability vector that approximates a sample from the categorical distribution of words at each step, and use the probability vector to compute the expected word embedding for the subsequent step.

Model	**28k**	**59k**	**120k**
LSTM[(a)]	57.9	62.5	65.9
CNN[(b)]	58.7	62.7	65.6
LSTM-AE[(a)]	59.9	64.6	68.5
LSTM-ADAE[(a)]	62.5	66.8	70.9
DeConv-AE[(b)]	62.1	65.5	68.7
LSTM-VAE[(b)]	64.7	67.5	71.1
DeConv-VAE[(b)]	67.2	69.3	72.2
LSTM-vMF-VAE (ours)	65.6	68.7	71.1
CS-LVM (ours)	68.4	73.5	76.9
$+\mathcal{R}^y$	**70.0**	**74.5**	77.4
$+\mathcal{R}^{\mathbf{z}}$	69.2	73.9	77.6
$+\mathcal{R}^{\boldsymbol{\mu}}$	69.1	74.0	**77.6**
$+\mathcal{R}^y, \mathcal{R}^{\mathbf{z}}, \mathcal{R}^{\boldsymbol{\mu}}$	69.6	74.1	77.4

Table 1: Semi-supervised classification results on the SNLI dataset. (a) Zhao et al. (2018); (b) Shen et al. (2018a).

Model	**1k**	**5k**	**10k**	**25k**
CNN[(a)]	56.3	59.2	63.8	68.9
LSTM-AE[(a)]	59.3	63.8	67.2	70.9
DeConv-AE[(a)]	60.2	65.1	67.7	71.6
LSTM-VAE[(a)]	62.9	67.6	69.0	72.4
DeConv-VAE[(a)]	65.1	69.4	70.5	73.7
LSTM-vMF-VAE (ours)	65.0	69.9	72.1	74.9
CS-LVM (ours)	**66.5**	71.1	74.6	76.9
$+\mathcal{R}^y$	66.4	70.8	74.5	77.5
$+\mathcal{R}^{\mathbf{z}}$	**66.5**	**71.3**	74.8	77.1
$+\mathcal{R}^{\boldsymbol{\mu}}$	66.4	71.2	**74.9**	77.4
$+\mathcal{R}^y, \mathcal{R}^{\mathbf{z}}, \mathcal{R}^{\boldsymbol{\mu}}$	66.3	**71.3**	74.7	**77.6**

Table 2: Semi-supervised classification results on the Quora Question Pairs dataset. (a) Shen et al. (2018a).

When multiple constraints are used, they are combined using the homoscedastic uncertainty weighting (Kendall et al., 2018):[8]

$$\mathcal{R} = \frac{1}{\sigma_1^2}\mathcal{R}^y + \frac{1}{\sigma_2^2}\mathcal{R}^{\mathbf{z}} + \frac{1}{\sigma_3^2}\mathcal{R}^{\boldsymbol{\mu}} + \log \sigma_1 + \log \sigma_2 + \log \sigma_3, \qquad (14)$$

where $\sigma_1, \sigma_2, \sigma_3$ are trainable scalar parameters. Also note that all constraints are *unsupervised*, where label information is not required.

4 Experiments

We evaluate the proposed model on two semi-supervised tasks: *natural language inference* and *paraphrase identification*. We also implement a strong baseline that has a similar architecture to LSTM-VAE (Shen et al., 2018a) but uses vMF distribution for prior and posterior, named LSTM-vMF-VAE. To further explore the proposed model,

[7] The basic assumption behind this constraint is that a source and a target sequence are associated in a certain aspect, and it generally holds in most of the available pair classification datasets e.g. SNLI, SICK, SciTail, QQP, MRPC.

[8] Though the weighting scheme is originally derived from the case of a Gaussian likelihood, Kendall et al. (2018); Xiong et al. (2018); Hu et al. (2018) successfully applied it in weighting various losses e.g. cross-entropy loss, L_1 loss, and reinforcement learning objectives.

we conduct extensive qualitative analyses. For detailed settings and hyperparameters, please refer to Appendix C.

4.1 Natural Language Inference

Natural language inference (NLI) is a task of predicting the relationship given a premise and a hypothesis sentence. We use Stanford Natural Language Inference (SNLI, Bowman et al., 2015) dataset for experiments. It consists of roughly 570k premise-hypothesis pairs, and each pair has one of the following labels: *entailment*, *neutral*, and *contradiction*. Considering the asymmetry 不对称 in some label classes and for conformance 顺应 with the dataset generation process, we use premise and hypothesis sentence as source and target respectively: $(\mathbf{x}_s, \mathbf{x}_t) = (\mathbf{x}_{pre}, \mathbf{x}_{hyp})$.

Following the work of Zhao et al. (2018); Shen et al. (2018a), we consider scenarios where 28k, 59k, and 120k labeled data samples are available. Also, for fair comparison with the prior work, we set the size of a word vocabulary set to 20,000 and do not utilize pre-trained word embeddings such as GloVe (Pennington et al., 2014).

To combine the representations of a premise and a hypothesis and to construct an input to f^{disc}, we use the following heuristic-based fusion proposed by Mou et al. (2016):

$$\begin{aligned} & f^{fuse}(\mathbf{h}_{pre}, \mathbf{h}_{hyp}) \\ & \quad = [\mathbf{h}_{pre}; \mathbf{h}_{hyp}; |\mathbf{h}_{pre} - \mathbf{h}_{hyp}|; \mathbf{h}_{pre} \odot \mathbf{h}_{hyp}], \end{aligned} \tag{15}$$

where $[\mathbf{a}; \mathbf{b}]$ indicates concatenation of vectors $\mathbf{a}$, $\mathbf{b}$ and $\odot$ is the element-wise product.

Table 1 summarizes the result of experiments. We can clearly see that the proposed CS-LVM architecture substantially outperforms other models based on auto-encoding. Also, the semantic constraints brought additional boost in performance, achieving the new state of the art in semi-supervised classification of the SNLI dataset.

When all training data are used as labeled data ($\approx$ 550k), CS-LVM also improves performance by achieving accuracy of 82.8%, compared to the supervised LSTM (81.5%), LSTM-AE (81.6%), LSTM-VAE (80.8%), DeConv-VAE (80.9%).

4.2 Paraphrase Identification

Paraphrase identification (PI) is a task whose objective is to infer whether two sentences have the same semantics. We use the Quora Question Pairs dataset (QQP, Wang et al., 2017) for experiments. QQP consists of over 400k sentence pairs each of which has label information indicating whether the sentences in a pair paraphrase each other or not. We experiment for the cases where the number of labeled data is 1k, 5k, 10k, and 25k, and set the vocabulary size to 10,000, following Shen et al. (2018a). Unlike auto-encoding–based models that treat sentences in a pair equivalently, the CS-LVM processes them asymmetrically for its cross-sentence generating property. This property is useful when some relationships are asymmetric (e.g. NLI), however the paraphrase relationship is bidirectional, so that we also use swapped text pairs in training. To fuse sentence representations, the following symmetric function is used, as in Ji and Eisenstein (2013):

$$f^{fuse}(\mathbf{h}_1, \mathbf{h}_2) = [\mathbf{h}_1 + \mathbf{h}_2; |\mathbf{h}_1 - \mathbf{h}_2|]. \tag{16}$$

Model	28k	59k	120k
CS-LVM	68.4	73.5	76.9
(i) without CS	65.6	68.7	71.1
(ii) Gaussian	66.9	72.0	74.9
(iii) sampling	68.0	72.9	76.5
(iv) $f^{enc} \neq g^{enc}$	63.3	69.1	74.7

Table 3: Ablation study results.

The result of experiments on QQP is summarized in Table 2. Again, the proposed CS-LVM consistently outperforms other supervised and semi-supervised models by a large margin, setting the new state-of-the-art result on the QQP dataset with the semi-supervised setting.

4.3 Ablation Study

To assess the effect of each element, we experiment with model variants where some of the components are removed. Specifically, we conduct an ablation study for the following variants: (i) without cross-sentence generation (i.e. auto-encoding setup), (ii) replacing the vMF distribution with Gaussian, (iii) computing the expectation term of Eq. 9 by sampling, and (iv) without encoder weight sharing (i.e. $f^{enc} \neq g^{enc}$). SNLI dataset is used for the model ablation experiments, and trained models are not fine-tuned in order to focus only on the efficacy of each model component.

Results of ablation study are presented in Table 3. As expected, the cross-sentence generation is the most critical factor for the performance, except for the 28k setting where the encoder weight tying brought the biggest gain. In

59k and 120k settings, all other variants that maintain the cross-generating property outperform the VAE-based models (see *(ii)*, *(iii)*, *(iv)*).

Replacing a vMF with a Gaussian does not severely harm the accuracy, however it requires the additional process of finding a KL cost annealing rate. When sampling is used instead of enumeration for computing Eq. 9, about 1.2x speedup is observed in exchange for slight performance degradation, and thus sampling could be a good option in the case that the number of label classes is large.

Finally, as mentioned in §3.2, variants whose encoder weights are untied do not work well. We conjecture this is because g^{enc} receives the error signal only from a source sentence and could not fully benefit from both sentences. The fact that the performance degradation is larger when the number of labeled data is small also agrees with our hypothesis, since unlabeled data affect the classifier encoder only by the entropy term when encoder weights are not shared.

4.4 Generated Sentences

We give examples of generated sentences, to validate that the proposed model learns to generate text having desired properties. From Table 4, we can see that sentences generated from the identical input sentence properly reflect the label information given. More generated examples are presented in Appendix D.

Further, to quantitatively measure the quality of generated sentences, we construct artificial datasets, where each premise and label in the SNLI development set is used as input to our trained generator and generated hypotheses are collected. Then we prepare a LSTM classifier that is trained on the original SNLI dataset as a surrogate for the ideal classifier, and use it for measuring the quality of generated datasets.[9] We also compute the diversity of the generated hypotheses using the metrics proposed by Li et al. (2016), to verify the effect of diversity-promoting semantic constraints.

Results of the evaluation on the artificial datasets are presented in Table 5. The classifier trained on the original dataset predicts the generated data fairly well, from which we verify that the generated sentences contain desired semantics. Also, as expected, fine-tuning with $\mathcal{R}^y$ increases the classification accuracy by a large margin, while $\mathcal{R}^{\mathbf{z}}$ and $\mathcal{R}^{\boldsymbol{\mu}}$ enhance diversity.

5 Related Work

Semi-supervised learning for text classification. Using unlabeled data for text classification is an important subject and there exists much previous research (Zhu et al., 2003; Nigam et al., 2006; Zhu, 2008, to name but a few). Notably, the work of Xu et al. (2017) applies the semi-supervised VAE (Kingma et al., 2014) to the single-sentence text classification problem. Zhao et al. (2018); Shen et al. (2018a) present VAE models for the semi-supervised text sequence matching, while their models have drawbacks as mentioned in §3.

When the use of external corpora is allowed, the performance can further be increased. Dai and Le (2015); Ramachandran et al. (2017) train an encoder-decoder network on large corpora and fine-tune the learned encoder on a specific task. Recently, there have been remarkable improvements in pre-trained language representations (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2018), where language models trained on extremely large data brought a huge performance boost. These methods are orthogonal to our work, and additional enhancements are expected when they are used together with our model.

Cross-sentence generating LVMs. There exists some prior work on cross-sentence generating LVMs. Shen et al. (2017) introduce a similar data generation assumption to ours and apply the idea to unaligned style transfer and natural language generation. Zhang et al. (2016); Serban et al. (2017) use latent variable models for machine translation and dialogue generation. Kang et al. (2018) propose a data augmentation framework for natural language inference that generates a sentence, however unlabeled data are not considered in their work. Deudon (2018) build a sentence-reformulating deep generative model whose objective is to measure the semantic similarity between a sentence pair. However their work cannot be applied to a multi-class classification problem, and the generative objective is only used in pre-training, not considering the joint optimization of the generative and the discriminative objective. To the best of our knowledge, our work is the first work on introducing the concept of cross-sentence generating LVM to the semi-supervised text matching problem.

[9] The accuracy of the trained classifier on the original development set is 81.7%.

Input	**Entailment**	**Neutral**	**Contradiction**
two girls play with bubbles near a boat dock .	two girls are outside .	the girls are friends .	two girls are swimming in the ocean .
a classroom full of men, with the teacher up front .	a group of boys are indoors .	the teacher is teaching the students .	the students are at home sleeping .
a dune buggy traveling on sand .	the vehicle is moving .	the vehicle is red .	a man is riding a bike .

Table 4: Selected samples generated from the model trained on the SNLI dataset.

Dataset	**Acc.**	***distinct-1***	***distinct-2***
CS-LVM	76.5	.0128	.0441
$+\mathcal{R}^y$	81.9	.0135	.0479
$+\mathcal{R}^{\mathbf{z}}$	79.0	.0140	.0492
$+\mathcal{R}^{\boldsymbol{\mu}}$	77.5	.0141	.0488

Table 5: Results of evaluation of generated artificial datasets. *distinct-1* and *distinct-2* compute the ratio of the number of unique unigrams or bigrams to that of the total generated tokens (Li et al., 2016).

6 Conclusion

In this work, we proposed a cross-sentence latent variable model (CS-LVM) for semi-supervised text sequence matching. Given a pair of text sequences and the corresponding label, it uses one of the sequences and the label as input and generates the other sequence. Due to the use of cross-sentence generation, the generative model and the discriminative classifier interacts more strongly, and from experiments we empirically proved that the CS-LVM outperforms other models by a large margin. We also defined multiple semantic constraints to further regularize the model, and observed that fine-tuning with them gives additional increase in performance.

For future work, we plan to focus on generating more realistic text and use the generated text in other tasks e.g. data augmentation, addressing adversarial attack. Although the current model makes fairly plausible sentences, it tends to prefer relatively short and *safe* sentences, as the main goal of the training is to accurately predict the relationship between sentences. We expect the model could perform more natural generation via applying recent advancements on deep generative models.

Acknowledgments

This work was supported by BK21 Plus for Pioneers in Innovative Computing (Dept. of Computer Science and Engineering, SNU) funded by the National Research Foundation of Korea (NRF) (21A20151113068). We would also like to thank anonymous reviewers for their constructive feedbacks.

References

Alexander Alemi, Ben Poole, Ian Fischer, Joshua Dillon, Rif A. Saurous, and Kevin Murphy. 2018. Fixing a broken ELBO. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 159–168, Stockholm, Sweden. PMLR.

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

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. 2016a. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1466–1477, Berlin, Germany. Association for Computational Linguistics.

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

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. 2010. *Semi-Supervised Learning*. Adaptive computation and machine learning. MIT Press.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2406–2417, Melbourne, Australia. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017a. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2017b. Variational lossy autoencoder. In *International Conference on Learning Representations*, Toulon, France.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems 28*, pages 3079–3087, Montreal, Canada. Curran Associates, Inc.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. 2018. Hyperspherical variational auto-encoders. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 856–865, Monterey, California, USA.

Michel Deudon. 2018. Learning semantic similarity in a continuous space. In *Advances in Neural Information Processing Systems 31*, pages 986–997, Montreal, Canada. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository*, arXiv:1810.04805. Version 1.

Adji B. Dieng, Yoon Kim, Alexander M. Rush, and David M. Blei. 2019. Avoiding latent variable collapse with generative skip models. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, Naha, Japan. PMLR.

Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. 2017. PixelVAE: A latent variable model for natural images. In *International Conference on Learning Representations*, Toulon, France.

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.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and Ming Zhou. 2018. Reinforced mnemonic reader for machine reading comprehension. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 4099–4106, Stockholm, Sweden. International Joint Conferences on Artificial Intelligence Organization.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2017. Tying word vectors and word classifiers: A loss framework for language modeling. In *International Conference on Learning Representations*, Toulon, France.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, Toulon, France.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896, Seattle, Washington, USA. Association for Computational Linguistics.

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. 2018. AdvEntuRe: Adversarial training for textual entailment with knowledge-guided examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2418–2428, Melbourne, Australia. Association for Computational Linguistics.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, Salt Lake City, Utah, USA.

Taeuk Kim, Jihun Choi, Daniel Edmiston, Sanghwan Bae, and Sang-goo Lee. 2019. Dynamic compositionality in recursive neural networks with structure-aware tag representations. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, Honolulu, Hawaii, USA.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, Banff, Canada.

Durk P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems 27*, pages 3581–3589, Montreal, Canada. Curran Associates, Inc.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California, USA. Association for Computational Linguistics.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, Toulon, France.

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.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 130–136, Berlin, Germany. Association for Computational Linguistics.

Kamal Nigam, Andrew McCallum, and Tom Mitchell. 2006. Semi-supervised text classification using EM. *Semi-Supervised Learning*, pages 33–56.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems 30*, pages 6306–6315, Long Beach, California, USA. Curran Associates, Inc.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. 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*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

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, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163, Valencia, Spain. Association for Computational Linguistics.

Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. 2016. Variational autoencoder for deep learning of images, labels and captions. In *Advances in Neural Information Processing Systems 29*, pages 2352–2360, Barcelona, Spain. Curran Associates, Inc.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Technical report, OpenAI.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, USA. Association for Computational Linguistics.

Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391, Copenhagen, Denmark. Association for Computational Linguistics.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Beijing, China. PMLR.

Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. 2018. A hierarchical latent vector model for learning long-term structure in music. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4364–4373, Stockholm, Sweden. PMLR.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3295–3301, San Francisco, California, USA.

Dinghan Shen, Yizhe Zhang, Ricardo Henao, Qinliang Su, and Lawrence Carin. 2018a. Deconvolutional latent-variable model for text sequence matching. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5438–5445, New Orleans, Louisiana, USA.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018b. DiSAN: Directional self-attention network for RNN/CNN-free language understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5446–5455, New Orleans, Louisiana, USA.

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 30*, pages 6830–6841, Long Beach, California, USA. Curran Associates, Inc.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems 28*, pages 3483–3491, Montreal, Canada. Curran Associates, Inc.

Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 464–473, Berlin, Germany. Association for Computational Linguistics.

Kateryna Tymoshenko and Alessandro Moschitti. 2018. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173, Brussels, Belgium. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *International Conference on Learning Representations*, Toulon, France.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 4144–4150, Melbourne, Australia.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Computing Research Repository*, arXiv:1609.08144. Version 2.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. DCN+: Mixed objective and deep residual coattention for question answering. In *International Conference on Learning Representations*, Vancouver, Canada.

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, Brussels, Belgium. Association for Computational Linguistics.

Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 3358–3364, San Francisco, California, USA.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. 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, 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, Austin, Texas, USA. Association for Computational Linguistics.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5902–5911, Stockholm, Sweden. PMLR.

Xiaojin Zhu. 2008. Semi-supervised learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin-Madison.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, pages 912–919. AAAI Press.

A von Mises–Fisher Distribution

A von Mises–Fisher (vMF) distribution is the distribution defined on a m-dimensional unit hypersphere. It is parameterized by two parameters: the mean direction $\boldsymbol{\mu} \in \mathbb{R}^m$ and the concentration $\kappa \in \mathbb{R}$. The probability density function (pdf) of vMF$(\boldsymbol{\mu}, \kappa)$ is defined by

$$f(\mathbf{x}; \boldsymbol{\mu}, \kappa) = C_m(\kappa) \exp(\kappa \boldsymbol{\mu}^\top \mathbf{x}), \tag{17}$$

where

$$C_m(\kappa) = \frac{\kappa^{m/2-1}}{(2\pi)^{m/2} I_{m/2-1}(\kappa)}$$

and $I_v(\kappa)$ is the modified Bessel function of the first kind at order v. Eq. 17 is used in the computation of $\mathcal{R}^{\mathbf{z}}$.

A sample from a vMF distribution is drawn from the acceptance-rejection scheme presented in Algorithm 1 of Davidson et al. (2018). In their algorithm, a stochastic variable obtained from the acceptance-rejection sampling does not depend on μ, thus the sampling process can be rewritten as a deterministic function that accepts the stochastic variable as input (i.e. reparameterization trick).

The KL divergence between a vMF distribution vMF$(\boldsymbol{\mu}, \kappa)$ and the hyperspherical uniform distribution $\mathcal{U}(S^{m-1}) = \text{vMF}(\cdot, 0)$ can be derived analytically:

$$\begin{aligned}
&D_{KL}(\text{vMF}(\boldsymbol{\mu}, \kappa) \| \text{vMF}(\cdot, 0)) \\
&= \log C_m(\kappa) - \log \frac{\Gamma(m/2)}{2\pi^{m/2}} + \kappa \frac{I_{m/2}(\kappa)}{I_{m/2-1}(\kappa)}.
\end{aligned}$$

Note that the KL divergence does not depend on $\boldsymbol{\mu}$, thus the KL divergence is a constant if κ is fixed. Intuitively, this is because the hyperspherical uniform distribution has equal probability density at every point on the unit hypersphere, and $D_{KL}(\text{vMF}(\boldsymbol{\mu}, \kappa) \| \text{vMF}(\cdot, 0))$ should not be changed under rotations.

B Derivation of Lower Bounds

Let $q_{\boldsymbol{\theta}}(\mathbf{z}_s|\cdot)$ be a distribution that has the same support with $p(\mathbf{z}_s)$. Then the KL divergence between $q_{\boldsymbol{\theta}}(\mathbf{z}_s|\cdot)$ and $p_{\boldsymbol{\theta}}(\mathbf{z}_s|\mathbf{x}_t, y)$ can be written as

$$\begin{aligned}
&D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot) \| p_{\boldsymbol{\theta}}(\mathbf{z}_s|\mathbf{x}_t, y)) \\
&\quad = \int q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot) \log \frac{q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot)}{p_{\boldsymbol{\theta}}(\mathbf{z}_s|\mathbf{x}_t, y)} d\mathbf{z}_s \\
&\quad = \int q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot) \log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t, y) q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot)}{p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y)} d\mathbf{z}_s \\
&\quad = \log p_{\boldsymbol{\theta}}(\mathbf{x}_t, y) + D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot) \| p(\mathbf{z}_s)) \\
&\qquad - \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}_s|\cdot)}[\log p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y)] - \log p(y) \\
&\quad \geq 0.
\end{aligned}$$

From the above inequality we obtain the lower bound of $\log p_{\boldsymbol{\theta}}(\mathbf{x}_t, y)$ presented in Eq. 2.

The lower bound of $\log p_{\boldsymbol{\theta}}(\mathbf{x}_t)$ (Eq. 7) could be derived as follows.

$$\begin{aligned}
&\log p_{\boldsymbol{\theta}}(\mathbf{x}_t) = \log \sum_y \int p_{\boldsymbol{\theta}}(\mathbf{x}_t, \mathbf{z}_s, y) d\mathbf{z}_s \\
&= \log \mathbb{E}_{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \left[\frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y)}{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \right] \\
&\geq \mathbb{E}_{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y)}{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \right]
\end{aligned}$$

From the graphical model $q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t) = q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s) q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t)$, and thus

$$\begin{aligned}
&\mathbb{E}_{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y)}{q_{\boldsymbol{\phi},\boldsymbol{\psi}}(y, \mathbf{z}_s|\mathbf{x}_s, \mathbf{x}_t)} \right] \\
&\quad = \mathbb{E}_{q_{\boldsymbol{\psi}}} \left[\mathbb{E}_{q_{\boldsymbol{\phi}}} \left[\log \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{z}_s, y) p(\mathbf{z}_s) p(y)}{q_{\boldsymbol{\phi}}(\mathbf{z}_s|\mathbf{x}_s)} \right] \right] \\
&\qquad\qquad - \mathbb{E}_{q_{\boldsymbol{\psi}}}[\log q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t)] \\
&\quad = \mathbb{E}_{q_{\boldsymbol{\psi}}} \left[-\mathcal{L}_l^{gen}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}_s, \mathbf{x}_t, y) \right] \\
&\qquad\qquad + \mathcal{H}(q_{\boldsymbol{\psi}}(y|\mathbf{x}_s, \mathbf{x}_t)) \\
&\quad = -\mathcal{L}_u(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\psi}; \mathbf{x}_s, \mathbf{x}_t).
\end{aligned}$$

C Implementation Details

We used PyTorch[10] and AllenNLP[11] libraries for implementation. The default weight initialization scheme of the AllenNLP library is used unless explicitly stated.

For all CS-LVM experiments, the size of word embeddings and hidden dimensions of LSTMs are set to 300, and the size of label embeddings is 50. g^{code} is implemented as a linear projection of the last hidden state of the encoder LSTM followed by normalization. g^{out} is a linear projection followed by the softmax function, and we reuse the word embeddings as its weight matrix (Press and Wolf, 2017; Inan et al., 2017). The discriminative classifier is a feedforward network with single hidden layer and the ReLU activation function, and the hidden dimension is set to 1200. We apply dropout on word embeddings and the classifier with probabilities p_w and p_c respectively.

When multiple semantic constraints are used, to make uncertainty weights be always positive and be optimized stably, we instead use $\log \sigma_i^2$ as model parameter, as in Kendall et al. (2018). Each $\log \sigma_i^2$ is initialized with zero. The temperature parameter of the Gumbel-Softmax is linearly annealed using the following schedule:

$$\tau(t) = \max(0.1, 1.0 - rt),$$

[10] https://pytorch.org/

[11] https://allennlp.org/

Model	κ	λ	p_w	p_c
28k	150	0.8	0.75	0.1
59k	100	1.0	0.75	0.1
120k	120	0.8	0.50	0.1

Table 6: Hyperparameters for the SNLI models.

Model	κ	λ	p_w	p_c
1k	100	0.8	0.50	0.2
5k	120	0.5	0.75	0.2
10k	150	0.5	0.75	0.1
25k	100	0.5	0.75	0.1

Table 7: Hyperparameters for the QQP models.

where $r = 10^{-4}$ is the annealing rate and t is the training step.

To find optimal hyperparameters, we performed a rough grid search on $\kappa \in \{100, 120, 150\}$, $\lambda \in \{0.2, 0.5, 0.8, 1.0\}$, $p_w \in \{0.25, 0.50, 0.75\}$, and $p_c \in \{0.1, 0.2\}$. The KL divergence between a posterior and the prior is 23.57, 27.09, 31.60 when κ is set to 100, 120, 150 respectively.

For the LSTM-vMF-VAE experiments, we used the same hyperparameters and grid search scheme with those of the CS-LVM, except that we perform an additional search on the dimension of a latent code $d \in \{50, 150, 300\}$.

Adam optimizer (Kingma and Ba, 2015) with learning rate $\gamma = 10^{-3}$ is used for all experiments, except for 1k QQP experiments where stochastic gradient descent optimizer is used. When fine-tuning the model, we set γ to 10^{-4}. For other hyperparameters, we follow the configuration suggested by the authors. Best hyperparameter configurations found for SNLI and QQP datasets are presented in Tables 6 and 7.

D Generated Examples

We used beam search with $B = 10$ when generating sentences, and the length normalization (Wu et al., 2016) is applied with $\alpha = 0.7$.

Examples are presented in Tables 8–11. Though almost all generated hypotheses are realistic, we see that they lack diversity and fail to encode label information in some cases. For example, the phrase ‘is/are sleeping’ appears in generated sentences frequently when conditioned on the ‘contradiction’ label, likely because generating a set of simple patterns could be a shortcut to the objective. In Table 5, we verified from experiments that adding constraints helps enhancing accuracy and diversity, however a model is still relatively in favor of generating ‘easy’ sentences. We conjecture that the problem has its root in the fact that the primary objective of our model is to correctly classify the input, not to generate diverse outputs.

Input	**Entailment**	**Neutral**	**Contradiction**
little kids enjoy sprinklers by running through them outdoors .	kids are running .	the children are siblings .	the children are playing video games .
blurry people walking in the city at night .	people are walking outside .	the people are going to work .	the people are inside .
a woman sits in a chair under a tree and plays an acoustic guitar .	a woman is playing an instrument .	the woman is a musician .	a woman is playing the flute .
three men converse in a crowd .	three men are talking .	~~the men are talking .~~	the men are sleeping .
a woman in a yellow shirt seated at a table .	a woman is sitting .	~~a woman is sitting at a table .~~	the woman is standing .
a woman hugs a fluffy white dog .	a woman is holding a dog .	~~a woman is with her dog .~~	a woman is sleeping .
a crowd of people in colorful dresses .	people in costumes	the people are in a parade .	~~the people are sitting in a circle .~~
a clown making a balloon animal for a pretty lady .	~~a clown is entertaining a crowd .~~	the clown is entertaining a crowd .	the clown is sleeping .

Table 8: Sentences generated from the CS-LVM model trained on the SNLI dataset. Failure cases are denoted by ~~strikethrough~~ text.

Input	**Entailment**	**Neutral**	**Contradiction**
little kids enjoy sprinklers by running through them outdoors .	kids are playing outside .	the kids are playing in the water .	the kids are sleeping .
blurry people walking in the city at night .	people are walking .	the people are walking to work .	the people are inside .
a woman sits in a chair under a tree and plays an acoustic guitar .	a woman is playing music .	the woman is a musician .	a woman is sleeping .
three men converse in a crowd .	three men are talking .	three men are talking about politics .	the men are sleeping .
a woman in a yellow shirt seated at a table .	a woman is sitting .	a tall human sitting .	the woman is standing .
a woman hugs a fluffy white dog .	a woman is holding a dog .	the dog belongs to the woman .	the dog is black .
a crowd of people in colorful dresses .	people in costumes	the people are in a parade .	the people are sleeping .
a clown making a balloon animal for a pretty lady .	a clown is performing .	the clown is entertaining a crowd .	the clown is sleeping .

Table 9: Sentences generated from the CS-LVM + $\mathcal{R}^y$ model trained on the SNLI dataset. Note that failed examples in Table 8 are corrected due to the use of $\mathcal{R}^y$.

Input	**Entailment**	**Neutral**	**Contradiction**
little kids enjoy sprinklers by running through them outdoors .	~~kids are playing in water .~~	the kids are having fun .	the kids are sleeping .
blurry people walking in the city at night .	people are walking .	the people are walking to work .	the people are inside .
a woman sits in a chair under a tree and plays an acoustic guitar .	a woman is playing an instrument .	the woman is a musician .	a woman is playing the drums .
three men converse in a crowd .	three men are talking .	three men are talking about politics .	the men are sleeping .
a woman in a yellow shirt seated at a table .	a woman is sitting .	~~a woman is sitting at a table .~~	the woman is standing
a woman hugs a fluffy white dog .	a woman is holding a dog .	a woman is playing with her dog .	a woman is sleeping .
a crowd of people in colorful dresses .	people are wearing costumes .	the people are in a parade .	~~the people are sitting down .~~
a clown making a balloon animal for a pretty lady .	a clown performs .	~~the clown is a clown .~~	the clown is sleeping .

Table 10: Sentences generated from the CS-LVM + $\mathcal{R}^{\mathbf{z}}$ model trained on the SNLI dataset. Failure cases are denoted by ~~strikethrough~~ text.

Input	**Entailment**	**Neutral**	**Contradiction**
little kids enjoy sprinklers by running through them outdoors .	kids are playing outside .	the kids are having fun .	the kids are sleeping .
blurry people walking in the city at night .	people are walking .	the people are walking to work .	the people are inside .
a woman sits in a chair under a tree and plays an acoustic guitar .	a woman is playing an instrument .	the woman is a musician .	a woman is playing the piano .
three men converse in a crowd .	three men are talking .	three men are talking about politics .	the men are sleeping .
a woman in a yellow shirt seated at a table .	a woman is sitting .	~~a woman is sitting at a table .~~	the woman is standing
a woman hugs a fluffy white dog .	a woman is holding a dog .	the dog belongs to the woman .	a woman is petting a cat .
a crowd of people in colorful dresses .	people are dressed up .	the people are in a parade .	~~the people are sitting down .~~
a clown making a balloon animal for a pretty lady .	~~a clown is blowing bubbles .~~	~~the clown is a clown .~~	the clown is sleeping .

Table 11: Sentences generated from the CS-LVM + $\mathcal{R}^{\mu}$ model trained on the SNLI dataset. Failure cases are denoted by ~~strikethrough~~ text.