Multi-space Variational Encoder-Decoders for Semi-supervised Labeled Sequence Transduction

Chunting Zhou, Graham Neubig
Language Technologies Institute
Carnegie Mellon University
ctzhou,gneubig@cs.cmu.edu

Abstract

Labeled sequence transduction is a task of transforming one sequence into another sequence that satisfies desiderata specified by a set of labels. In this paper we propose *multi-space variational encoder-decoders*, a new model for labeled sequence transduction with semi-supervised learning. The generative model can use neural networks to handle both discrete and continuous latent variables to exploit various features of data. Experiments show that our model provides not only a powerful supervised framework but also can effectively take advantage of the unlabeled data. On the SIGMORPHON morphological inflection benchmark, our model outperforms single-model state-of-art results by a large margin for the majority of languages.[1]

1 Introduction

This paper proposes a model for labeled sequence transduction tasks, tasks where we are given an input sequence and a set of labels, from which we are expected to generate an output sequence that reflects the content of the input sequence and desiderata specified by the labels. Several examples of these tasks exist in prior work: using labels to moderate politeness in machine translation results (Sennrich et al., 2016), modifying the output language of a machine translation system (Johnson et al., 2016), or controlling the length of a summary in summarization (Kikuchi et al., 2016). In particular, however, we are motivated by the task of morphological reinflection (Cotterell et al.,

---

[1] An implementation of our model are available at https://github.com/violet-zct/MSVED-morph-reinflection.
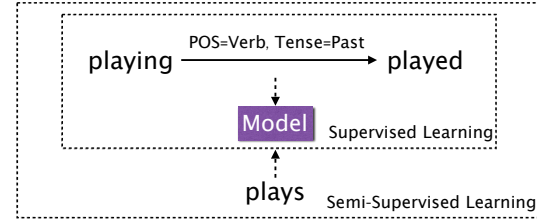

Figure 1: Standard supervised labeled sequence transduction, and our proposed semi-supervised method.

2016), which we will use as an example in our description and test bed for our models.

In morphologically rich languages, different affixes (i.e. prefixes, infixes, suffixes) can be combined with the lemma to reflect various syntactic and semantic features of a word. The ability to accurately analyze and generate morphological forms is crucial to creating applications such as machine translation (Chahuneau et al., 2013; Toutanova et al., 2008) or information retrieval (Darwish and Oard, 2007) in these languages. As shown in 1, *re*-inflection of an inflected form given the target linguistic labels is a challenging subtask of handling morphology as a whole, in which we take as input an inflected form (in the example, "playing") and labels representing the desired form ("pos=Verb, tense=Past") and must generate the desired form ("played").

Approaches to this task include those utilizing hand-crafted linguistic rules and heuristics (Taji et al., 2016), as well as learning-based approaches using alignment and extracted transduction rules (Durrett and DeNero, 2013; Alegria and Etxeberria, 2016; Nicolai et al., 2016). There have also been methods proposed using neural sequence-to-sequence models (Faruqui et al., 2016; Kann et al., 2016; Ostling, 2016), and currently ensembles of attentional encoder-decoder models (Kann and Schütze, 2016a,b) have achieved state-of-art results on this task. One feature of these neural models however, is that they are trained in a

largely *supervised* fashion (top of Fig. 1), using data explicitly labeled with the input sequence and labels, along with the output representation. Needless to say, the ability to obtain this annotated data for many languages is limited. However, we can expect that for most languages we can obtain large amounts of unlabeled surface forms that may allow for *semi-supervised* learning over this unlabeled data (entirety of Fig. 1).[2]

In this work, we propose a new framework for labeled sequence transduction problems: multi-space variational encoder-decoders (**MSVED**, §3.3). MSVEDs employ continuous or discrete latent variables belonging to multiple separate probability distributions[3] to explain the observed data. In the example of morphological reinflection, we introduce a vector of continuous random variables that represent the lemma of the source and target words, and also one discrete random variable for each of the labels, which are on the source or the target side.

This model has the advantage of both providing a powerful modeling framework for supervised learning, and allowing for learning in an unsupervised setting. For labeled data, we maximize the variational lower bound on the marginal log likelihood of the data and annotated labels. For unlabeled data, we train an auto-encoder to reconstruct a word conditioned on its lemma and morphological labels. While these labels are unavailable, a set of discrete latent variables are associated with each unlabeled word. Afterwards we can perform posterior inference on these latent variables and maximize the variational lower bound on the marginal log likelihood of data.

Experiments on the SIGMORPHON morphological reinflection task (Cotterell et al., 2016) find that our model beats the state-of-the-art for a single model in the majority of languages, and is particularly effective in languages with more complicated inflectional phenomena. Further, we find that semi-supervised learning allows for significant further gains. Finally, qualitative evaluation of lemma representations finds that our model is able to learn lemma embeddings that match with human intuition.

[2] Faruqui et al. (2016) have attempted a limited form of semi-supervised learning by re-ranking with a standard n-gram language model, but this is not integrated with the learning process for the neural model and gains are limited.
[3] Analogous to multi-space hidden Markov models (Tokuda et al., 2002)

2 Labeled Sequence Transduction

In this section, we first present some notations regarding labeled sequence transduction problems in general, then describe a particular instantiation for morphological reinflection.

Notation: Labeled sequence transduction problems involve transforming a source sequence $\mathbf{x}^{(s)}$ into a target sequence $\mathbf{x}^{(t)}$, with some labels describing the particular variety of transformation to be performed. We use discrete variables $y_1^{(t)}, y_2^{(t)}, \cdots, y_K^{(t)}$ to denote the labels associated with each target sequence, where K is the total number of labels. Let $\mathbf{y}^{(t)} = [y_1^{(t)}, y_2^{(t)}, \cdots, y_K^{(t)}]$ denote a vector of these discrete variables. Each discrete variable $y_k^{(t)}$ represents a categorical feature pertaining to the target sequence, and has a set of possible labels. In the later sections, we also use $\mathbf{y}^{(t)}$ and $y_k^{(t)}$ to denote discrete latent variables corresponding to these labels.

Given a source sequence $\mathbf{x}^{(s)}$ and a set of associated target labels $\mathbf{y}^{(t)}$, our goal is to generate a target sequence $\mathbf{x}^{(t)}$ that exhibits the features specified by $\mathbf{y}^{(t)}$ using a probabilistic model $p(\mathbf{x}^{(t)}|\mathbf{x}^{(s)}, \mathbf{y}^{(t)})$. The best target sequence $\hat{\mathbf{x}}^{(t)}$ is then given by:

$$\hat{\mathbf{x}}^{(t)} = \arg\max_{\mathbf{x}^{(t)}} p(\mathbf{x}^{(t)}|\mathbf{x}^{(s)}, \mathbf{y}^{(t)}). \qquad (1)$$

Morphological Reinflection Problem: In morphological reinflection, the source sequence $\mathbf{x}^{(s)}$ consists of the characters in an inflected word (e.g., "played"), while the associated labels $\mathbf{y}^{(t)}$ describe some linguistic features (e.g., $y_{\text{pos}}^{(t)} = $ Verb, $y_{\text{tense}}^{(t)} = $ Past) that we hope to realize in the target. The target sequence $\mathbf{x}^{(t)}$ is therefore the characters of the re-inflected form of the source word (e.g., "played") that satisfy the linguistic features specified by $\mathbf{y}^{(t)}$. For this task, each discrete variable $y_k^{(t)}$ has a set of possible labels (e.g. pos=V, pos=ADJ, etc) and follows a multinomial distribution.

3 Proposed Method

3.1 Preliminaries: Variational Autoencoder

As mentioned above, our proposed model uses probabilistic latent variables in a model based on neural networks. The variational autoencoder (Kingma and Welling, 2014) is an efficient way to handle (continuous) latent variables in neural
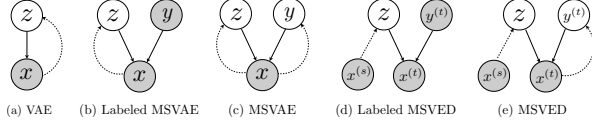
Figure 2: Graphical models of (a) VAE, (b) labeled MSVAE, (c) MSVAE, (d) labeled MSVED, and (e) MSVED. White circles are latent variables and shaded circles are observed variables. Dashed lines indicate the inference process while the solid lines indicate the generative process.

models. We describe it briefly here, and interested readers can refer to Doersch (2016) for details. The VAE learns a generative model of the probability $p(\mathbf{x}|\mathbf{z})$ of observed data $\mathbf{x}$ given a latent variable $\mathbf{z}$, and simultaneously uses a recognition model $q(\mathbf{z}|\mathbf{x})$ at learning time to estimate $\mathbf{z}$ for a particular observation $\mathbf{x}$ (Fig. 2(a)). $q(\cdot)$ and $p(\cdot)$ are modeled using neural networks parameterized by ϕ and θ respectively, and these parameters are learned by maximizing the variational lower bound on the marginal log likelihood of data:

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \\ \mathrm{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \quad (2)$$

The KL-divergence term (a standard feature of variational methods) ensures that the distributions estimated by the recognition model $q_\phi(\mathbf{z}|\mathbf{x})$ do not deviate far from our prior probability $p(\mathbf{z})$ of the values of the latent variables. To optimize the parameters with gradient descent, Kingma and Welling (2014) introduce a reparameterization trick that allows for training using simple backpropagation w.r.t. the Gaussian latent variables $\mathbf{z}$. Specifically, we can express $\mathbf{z}$ as a deterministic variable $\mathbf{z} = g_\phi(\epsilon, \mathbf{x})$ where ϵ is an independent Gaussian noise variable $\epsilon \sim \mathcal{N}(0, 1)$. The mean μ and the variance σ^2 of $\mathbf{z}$ are reparameterized by the differentiable functions w.r.t. ϕ. Thus, instead of generating $\mathbf{z}$ from $q_\phi(\mathbf{z}|\mathbf{x})$, we sample the auxiliary variable ϵ and obtain $\mathbf{z} = \mu_\phi(x) + \sigma_\phi(x) \circ \epsilon$, which enables gradients to backpropagate through ϕ.

3.2 Multi-space Variational Autoencoders

As an intermediate step to our full model, we next describe a generative model for a single sequence with both continuous and discrete latent variables, the multi-space variational auto-encoder (MSVAE). MSVAEs are a combination of two threads of previous work: deep generative models with both continuous/discrete latent variables for classification problems (Kingma et al., 2014;

Maaløe et al., 2016) and VAEs with only continuous variables for sequential data (Bowman et al., 2016; Chung et al., 2015; Zhang et al., 2016; Fabius and van Amersfoort, 2014; Bayer and Osendorfer, 2014). In MSVAEs, we have an observed sequence $\mathbf{x}$, continuous latent variables $\mathbf{z}$ like the VAE, as well as discrete variables $\mathbf{y}$.

In the case of the morphology example, $\mathbf{x}$ can be interpreted as an inflected word to be generated. $\mathbf{y}$ is a vector representing its linguistic labels, either annotated by an annotator in the observed case, or unannotated in the unobserved case. $\mathbf{z}$ is a vector of latent continuous variables, e.g. a latent embedding of the lemma that captures all the information about $\mathbf{x}$ that is not already represented in labels $\mathbf{y}$.

MSVAE: Because inflected words can be naturally thought of as "lemma+morphological labels", to interpret a word, we resort to discrete and continuous latent variables that represent the linguistic labels and the lemma respectively. In this case when the labels of the sequence $\mathbf{y}$ is not observed, we perform inference over possible linguistic labels and these inferred labels are referenced in generating $\mathbf{x}$.

The generative model $p_\theta(\mathbf{x}, \mathbf{y}, \mathbf{z}) = p(\mathbf{z})p_\pi(\mathbf{y})p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z})$ is defined as:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \quad (3)$$

$$p_\pi(\mathbf{y}) = \prod_k \mathrm{Cat}(y_k|\pi_k) \quad (4)$$

$$p_\theta(\mathbf{x}|\mathbf{y}, \mathbf{z}) = f(\mathbf{x}; \mathbf{y}, \mathbf{z}, \theta). \quad (5)$$

Like the standard VAE, we assume the prior of the latent variable $\mathbf{z}$ is a diagonal Gaussian distribution with zero mean and unit variance. We assume that each variable in $\mathbf{y}$ is independent, resulting in a factorized distribution in Eq. 4, where $\mathrm{Cat}(y_k|\pi_k)$ is a multinomial distribution with parameters π_k. For the purposes of this study, we set these to a uniform distribution $\pi_{k,j} = \frac{1}{|\pi_k|}$. $f(\mathbf{x}; \mathbf{y}, \mathbf{z}, \theta)$ calculates the likelihood of $\mathbf{x}$, a function parametrized by deep neural networks. Specifically, we employ an RNN decoder to generate the target word conditioned on the lemma variable $\mathbf{z}$ and linguistic labels $\mathbf{y}$, detailed in §5.

When inferring the latent variables from the given data $\mathbf{x}$, we assume the joint distribution of latent variables $\mathbf{z}$ and $\mathbf{y}$ has a factorized form, i.e. $q(\mathbf{z}, \mathbf{y}|\mathbf{x}) = q(\mathbf{z}|\mathbf{x})q(\mathbf{y}|\mathbf{x})$ as shown in Fig. 2(c).

The inference model is defined as follows:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \text{diag}(\sigma_\phi^2(\mathbf{x}))) \quad (6)$$

$$q_\phi(\mathbf{y}|\mathbf{x}) = \prod_k q_\phi(y_k|\mathbf{x})$$

$$= \prod_k \text{Cat}(y_k|\pi_\phi(\mathbf{x})) \quad (7)$$

where the inference distribution over $\mathbf{z}$ is a diagonal Gaussian distribution with mean and variance parameterized by neural networks. The inference model $q(\mathbf{y}|\mathbf{x})$ on labels $\mathbf{y}$ has the form of a discriminative classifier that generates a set of multinomial probability vectors $\pi_\phi(\mathbf{x})$ over all labels for each tag y_k. We represent each multinomial distribution $q(y_k|\mathbf{x})$ with an MLP.

The MSVAE is trained by maximizing the following variational lower bound $\mathcal{U}(\mathbf{x})$ on the objective for unlabeled data:

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{(\mathbf{y},\mathbf{z})\sim q_\phi(\mathbf{y},\mathbf{z}|\mathbf{x})} \log \frac{p_\theta(\mathbf{x},\mathbf{y},\mathbf{z})}{q_\phi(\mathbf{y},\mathbf{z}|\mathbf{x})}$$
$$= \mathbb{E}_{\mathbf{y}\sim q_\phi(\mathbf{y}|\mathbf{x})}[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z},\mathbf{y})]$$
$$- \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) + \log p_\pi(\mathbf{y})$$
$$- \log q_\phi(\mathbf{y}|\mathbf{x})] = \mathcal{U}(\mathbf{x}) \quad (8)$$

Note that this introduction of discrete variables requires more sophisticated optimization algorithms, which we will discuss in §4.1.

Labeled MSVAE: When $\mathbf{y}$ is observed as shown in Fig. 2(b), we maximize the following variational lower bound on the marginal log likelihood of the data and the labels:

$$\log p_\theta(\mathbf{x},\mathbf{y}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})} \log \frac{p_\theta(\mathbf{x},\mathbf{y},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} =$$
$$\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{y},\mathbf{z}) + \log p_\pi(\mathbf{y})]$$
$$- \text{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \quad (9)$$

which is a simple extension to Eq. 2.

Note that when labels are not observed, the inference model $q_\phi(\mathbf{y}|\mathbf{x})$ has the form of a discriminative classifier, thus we can use observed labels as the supervision signal to learn a better classifier. In this case we also minimize the following cross entropy as the classification loss:

$$\mathcal{D}(\mathbf{x},\mathbf{y}) = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim p_l(\mathbf{x},\mathbf{y})}[-\log q_\phi(\mathbf{y}|\mathbf{x})] \quad (10)$$

where $p_l(\mathbf{x},\mathbf{y})$ is the distribution of labeled data. This is a form of multi-task learning, as this additional loss also informs the learning of our representations.

3.3 Multi-space Variational Encoder-Decoders

Finally, we discuss the full proposed method: the multi-space variational encoder-decoder (MSVED), which generates the target $\mathbf{x}^{(t)}$ from the source $\mathbf{x}^{(s)}$ and labels $\mathbf{y}^{(t)}$. Again, we discuss two cases of this model: labels of the target sequence are observed and not observed.

MSVED: The graphical model for the MSVED is given in Fig. 2 (e). Because the labels of target sequence are not observed, once again we treat them as discrete latent variables and make inference on the these labels conditioned on the target sequence. The generative process for the MSVED is very similar to that of the MSVAE with one important exception: while the standard MSVAE conditions the recognition model $q(\mathbf{z}|\mathbf{x})$ on $\mathbf{x}$, then generates $\mathbf{x}$ itself, the MSVED conditions the recognition model $q(\mathbf{z}|\mathbf{x}^{(s)})$ on the source $\mathbf{x}^{(s)}$, then generates the target $\mathbf{x}^{(t)}$. Because only the recognition model is changed, the generative equations for $p_\theta(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}, \mathbf{z})$ are exactly the same as Eqs. 3–5 with $\mathbf{x}^{(t)}$ swapped for $\mathbf{x}$ and $\mathbf{y}^{(t)}$ swapped for $\mathbf{y}$. The variational lower bound on the conditional log likelihood, however, is affected by the recognition model, and thus is computed as:

$$\log p_\theta(\mathbf{x}^{(t)}|\mathbf{x}^{(s)})$$
$$\geq \mathbb{E}_{(\mathbf{y}^{(t)},\mathbf{z})\sim q_\phi(\mathbf{y}^{(t)},\mathbf{z}|\mathbf{x}^{(s)},\mathbf{x}^{(t)})} \log \frac{p_\theta(\mathbf{x}^{(t)},\mathbf{y}^{(t)},\mathbf{z}|\mathbf{x}^{(s)})}{q_\phi(\mathbf{y}^{(t)},\mathbf{z}|\mathbf{x}^{(s)},\mathbf{x}^{(t)})}$$
$$= \mathbb{E}_{\mathbf{y}^{(t)}\sim q_\phi(\mathbf{y}^{(t)}|\mathbf{x}^{(t)})}[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}^{(s)})}[\log p_\theta(\mathbf{x}^{(t)}|\mathbf{y}^{(t)},\mathbf{z})]$$
$$- \text{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(s)})||p(\mathbf{z})) + \log p_\pi(\mathbf{y}^{(t)})$$
$$- \log q_\phi(\mathbf{y}^{(t)}|\mathbf{x}^{(t)})] = \mathcal{L}_u(\mathbf{x}^{(t)}|\mathbf{x}^{(s)}) \quad (11)$$

Labeled MSVED: When the complete form of $\mathbf{x}^{(s)}$, $\mathbf{y}^{(t)}$, and $\mathbf{x}^{(t)}$ is observed in our training data, the graphical model of the labeled MSVED model is illustrated in Fig. 2 (d). We maximize the variational lower bound on the conditional log likelihood of observing $\mathbf{x}^{(t)}$ and $\mathbf{y}^{(t)}$ as follows:

$$\log p_\theta(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}|\mathbf{x}^{(s)})$$
$$\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}^{(s)})} \log \frac{p_\theta(\mathbf{x}^{(t)},\mathbf{y}^{(t)},\mathbf{z}|\mathbf{x}^{(s)})}{q_\phi(\mathbf{z}|\mathbf{x}^{(s)})}$$
$$= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x}^{(s)})}[\log p_\theta(\mathbf{x}^{(t)}|\mathbf{y}^{(t)},\mathbf{z}) + \log p_\pi(\mathbf{y}^{(t)})] -$$
$$\text{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(s)})||p(\mathbf{z})) = \mathcal{L}_l(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}|\mathbf{x}^{(s)}) \quad (12)$$

4 Learning MSVED

Now that we have described our overall model, we discuss details of the learning process that prove

useful to its success.

4.1 Learning Discrete Latent Variables

One challenge in training our model is that it is not trivial to perform back-propagation through discrete random variables, and thus it is difficult to learn in the models containing discrete tags such as MSVAE or MSVED.[4] To alleviate this problem, we use the recently proposed Gumbel-Softmax trick (Maddison et al., 2014; Gumbel and Lieblein, 1954) to create a differentiable estimator for categorical variables.

The Gumbel-Max trick (Gumbel and Lieblein, 1954) offers a simple way to draw samples from a categorical distribution with class probabilities $\pi_1, \pi_2, \cdots$ by using the `argmax` operation as follows: $\text{one\_hot}(\arg\max_i[g_i + \log \pi_i])$, where $g_1, g_2, \cdots$ are i.i.d. samples drawn from the Gumbel(0,1) distribution.[5] When making inferences on the morphological labels $y_1, y_2, \cdots$, the Gumbel-Max trick can be approximated by the continuous softmax function with temperature τ to generate a sample vector $\hat{\mathbf{y}}_i$ for each label i:

$$\hat{y}_{ij} = \frac{\exp((\log(\pi_{ij}) + g_{ij})/\tau)}{\sum_{k=1}^{N_i} \exp((\log(\pi_{ik}) + g_{ik})/\tau)} \quad (13)$$

where N_i is the number of classes of label i. When τ approaches zero, the generated sample $\hat{\mathbf{y}}_i$ becomes a one-hot vector. When $\tau > 0$, $\hat{\mathbf{y}}_i$ is smooth w.r.t π_i. In experiments, we start with a relatively large temperature and decrease it gradually.

4.2 Learning Continuous Latent Variables

MSVED aims at generating the target sequence conditioned on the latent variable $\mathbf{z}$ and the target labels $\mathbf{y}^{(t)}$. This requires the encoder to generate an informative representation $\mathbf{z}$ encoding the content of the $\mathbf{x}^{(s)}$. However, the variational lower bound in our loss function contains the KL-divergence between the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ and the prior $p(\mathbf{z})$, which is relatively easy to learn compared with learning to generate output from a latent representation. We observe that with the vanilla implementation the KL cost quickly decreases to near zero, setting $q_\phi(\mathbf{z}|\mathbf{x})$ equal to standard normal distribution. In

---

[4] Kingma et al. (2014) solve this problem by marginalizing over all labels, but this is infeasible in our case where we have an exponential number of label combinations.

[5] The Gumbel (0,1) distribution can be sampled by first drawing $u \sim \text{Uniform}(0,1)$ and computing $g = -\log(-\log(u))$.

this case, the RNN decoder can easily rely on the true output of last time step during training to decode the next token, which degenerates into an RNN language model. Hence, the latent variables are ignored by the decoder and cannot encode any useful information. The latent variable $\mathbf{z}$ learns an undesirable distribution that coincides with the imposed prior distribution but has no contribution to the decoder. To force the decoder to use the latent variables, we take the following two approaches which are similar to Bowman et al. (2016).

KL-Divergence Annealing: We add a coefficient λ to the KL cost and gradually anneal it from zero to a predefined threshold $\lambda_{\mathbf{m}}$. At the early stage of training, we set λ to be zero and let the model first figure out how to project the representation of the source sequence to a roughly right point in the space and then regularize it with the KL cost. Although we are not optimizing the tight variational lower bound, the model balances well between generation and regularization. This technique can also be seen in (Kočiskỳ et al., 2016; Miao and Blunsom, 2016).

Input Dropout in the Decoder: Besides annealing the KL cost, we also randomly drop out the input token with a probability of β at each time step of the decoder during learning. The previous ground-truth token embedding is replaced with a zero vector when dropped. In this way, the RNN decoder could not fully rely on the ground-truth previous token, which ensures that the decoder uses information encoded in the latent variables.

5 Architecture for Morphological Reinflection

Training details: For the morphological reinflection task, our supervised training data consists of source $\mathbf{x}^{(s)}$, target $\mathbf{x}^{(t)}$, and target tags $\mathbf{y}^{(t)}$. We test three variants of our model trained using different types of data and different loss functions. First, the single-directional supervised model (**SD-Sup**) is purely supervised: it only decodes the target word from the given source word with the loss function $\mathcal{L}_l(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}|\mathbf{x}^{(s)})$ from Eq. 12. Second, the bi-directional supervised model (**BD-Sup**) is trained in both directions: decoding the target word from the source word and decoding the source word from the target word, which corresponds to the loss function $\mathcal{L}_l(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}|\mathbf{x}^{(s)}) + \mathcal{L}_u(\mathbf{x}^{(s)}|\mathbf{x}^{(t)})$ using Eqs. 11–12. Finally, the semi-supervised model (**Semi-sup**) is trained to maxi-

Language	#LD	#ULD	Proposed MSVED			Baseline MED	
			SD-Sup	BD-Sup	Semi-sup	Single	Ensemble
Turkish	12,798	29,608	93.25	95.66[†]	**97.25[‡]**	89.56	95.00
Hungarian	19,200	34,025	97.00	98.54[†]	**99.16[‡]**	96.46	98.37
Spanish	12,799	72,151	88.32	91.50	93.74	94.74[†‡]	**96.69**
Russian	12,798	67,691	75.77	83.07	86.80[‡]	83.55[†]	**87.13**
Navajo	12,635	6,839	85.00	95.37[†]	**98.25[‡]**	63.62	83.00
Maltese	19,200	46,918	84.83	88.21[†]	**88.46[‡]**	79.59	84.25
Arabic	12,797	53,791	79.13	92.62[†]	**93.37[‡]**	72.58	82.80
Georgian	12,795	46,562	89.31	93.63[†]	95.97[‡]	91.06	**96.21**
German	12,777	56,246	75.55	84.08	90.28[‡]	89.11[†]	**92.41**
Finnish	12,800	74,687	75.59	85.11	91.20[‡]	85.63[†]	**93.18**
Avg. Acc	–	–	84.38	90.78[†]	**93.45[‡]**	84.59	90.90

Table 1: Results for Task 3 of SIGMORPHON 2016 on Morphology Reinflection. † represents the best single supervised model score, ‡ represents the best model including semi-supervised models, and bold represents the best score overall. **#LD** and **#ULD** are the number of supervised data and unlabeled words respectively.
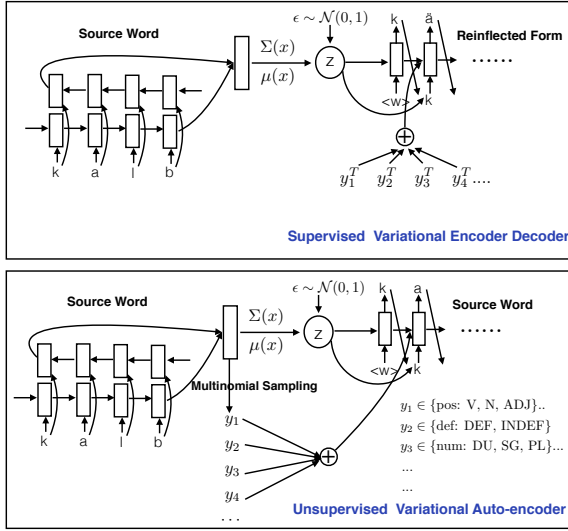

Figure 3: Model architecture for labeled and unlabeled data. For the encoder-decoder model, only one direction from the source to target is given. The classification model is not illustrated in the diagram.

mize the variational lower bounds and minimize the classification cross-entropy error of 10.

$$\mathcal{L}(\mathbf{x^{(s)}}, \mathbf{x^{(t)}}, \mathbf{y^{(t)}}, \mathbf{x}) = \alpha \cdot \mathcal{U}(\mathbf{x}) + \mathcal{L}_u(\mathbf{x}^{(s)}|\mathbf{x}^{(t)})$$
$$+ \mathcal{L}_l(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}|\mathbf{x}^{(s)}) - \mathcal{D}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \quad (14)$$

The weight α controls the relative weight between the loss from unlabeled data and labeled data.

We use Monte Carlo methods to estimate the expectation over the posterior distribution $q(\mathbf{z}|\mathbf{x})$ and $q(\mathbf{y}|\mathbf{x})$ inside the objective function 14. Specifically, we draw Gumbel noise and Gaussian noise one at a time to compute the latent variables $\mathbf{y}$ and $\mathbf{z}$.

The overall model architecture is shown in Fig. 3. Each character and each label is associated with a continuous vector. We employ Gated Recurrent Units (GRUs) for the encoder and de-

coder. Let $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ denote the hidden state of the forward and backward encoder RNN at time step t. $\mathbf{u}$ is the hidden representation of $\mathbf{x}^{(s)}$ concatenating the last hidden state from both directions i.e. $[\overrightarrow{h_T}; \overleftarrow{h_T}]$ where T is the word length. $\mathbf{u}$ is used as the input for the inference model on $\mathbf{z}$. We represent $\mu(\mathbf{u})$ and $\sigma^2(\mathbf{u})$ as MLPs and sample $\mathbf{z}$ from $\mathcal{N}(\mu(\mathbf{u}), \mathrm{diag}(\sigma^2(\mathbf{u})))$, using $\mathbf{z} = \mu + \sigma \circ \epsilon$, where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Similarly, we can obtain the hidden representation of $\mathbf{x}^{(t)}$ and use this as input to the inference model on each label $\mathbf{y}_i^{(t)}$ which is also an MLP following a softmax layer to generate the categorical probabilities of target labels.

In decoding, we use 3 types of information in calculating the probability of the next character : (1) the current decoder state, (2) a tag context vector using attention (Bahdanau et al., 2015) over the tag embeddings, and (3) the latent variable $\mathbf{z}$. The intuition behind this design is that we would like the model to constantly consider the lemma represented by $\mathbf{z}$, and also reference the tag corresponding to the current morpheme being generated at this point. We do not marginalize over the latent variable $\mathbf{z}$ however, instead we use the mode μ of $\mathbf{z}$ as the latent representation for $\mathbf{z}$. We use beam search with a beam size of 8 to perform search over the character vocabulary at each decoding time step.

Other experimental setups: All hyperparameters are tuned on the validation set, and include the following: For KL cost annealing, λ_m is set to be 0.2 for all language settings. For character drop-out at the decoder, we empirically set β to be 0.4 for all languages. We set the dimension of character embeddings to be 300, tag label embeddings to be 200, RNN hidden state to be 256, and

latent variable **z** to be 150. We set α the weight for the unsupervised loss to be 0.8. We train the model with Adadelta (Zeiler, 2012) and use early-stop with a patience of 10.

6 Experiments

6.1 Background: SIGMORPHON 2016

SIGMORPHON 2016 is a shared task on morphological inflection over 10 different morphologically rich languages. There are a total of three tasks, the most difficult of which is task 3, which requires the system to output the reinflection of an inflected word.[6] The training data format in task 3 is in triples: (source word, target labels, target word). In the test phase, the system is asked to generate the target word given a source word and the target labels. There are a total of three tracks for each task, divided based the amount of supervised data that can be used to solve the problem, among which track 2 has the strictest limitation of only using data for the corresponding task. As this is an ideal testbed for our method, which can learn from unlabeled data, we choose track 2 and task 3 to test our our model's ability to exploit this data.

As a baseline, we compare our results with the MED system (Kann and Schütze, 2016a) which achieved state-of-the-art results in the shared task. This system used an encoder-decoder model with attention on the concatenated source word and target labels. Its best result is obtained from an ensemble of five RNN encoder-decoders (**Ensemble**). To make a fair comparison with our models, which don't use ensembling, we also calculated single model results (**Single**).

All models are trained using the labeled training data provided for task 3. For our semi-supervised model (**Semi-sup**), we also leverage unlabeled data from the training and validation data for tasks 1 and 2 to train variational auto-encoders.

6.2 Results and Analysis

From the results in Tab. 1, we can glean a number of observations. First, comparing the results of our full **Semi-sup** model, we can see that for all languages except Spanish, it achieves accuracies better than the single MED system, often by a large margin. Even compared to the MED ensembled model, our single-model system is quite competitive, achieving higher accuracies for Hungarian,

---

[6]Task 1 is inflection of a lemma word and task 2 is reinflection but also provides the source word labels.

Language	Prefix	Stem	Suffix
Turkish	0.21	1.12	98.75
Hungarian	0.00	0.08	99.79
Spanish	0.09	3.25	90.74
Russian	0.66	7.70	85.00
Navajo	77.64	18.38	26.40
Maltese	48.81	11.05	98.74
Arabic	68.52	37.04	88.24
Georgian	4.46	0.41	92.47
German	0.84	3.32	89.19
Finnish	0.02	12.33	96.16

Table 2: Percentage of inflected word forms that have modified each part of the lemma (Cotterell et al., 2016) (some words can be inflected zero or multiple times, thus sums may not add to 100%).
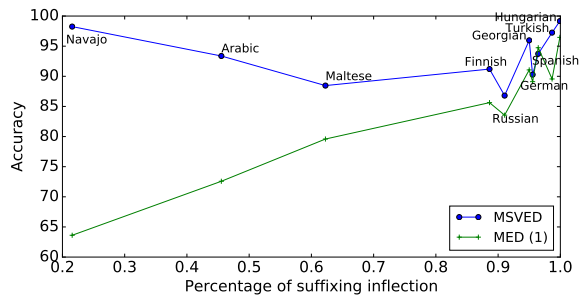

Figure 4: Performance on test data w.r.t. the percentage of suffixing inflection. Points with the same x-axis value correspond to the same language results.

Navajo, Maltese, and Arabic, as well as achieving average accuracies that are state-of-the-art.

Next, comparing the different varieties of our proposed models, we can see that the semi-supervised model consistently outperforms the bidirectional model for all languages. And similarly, the bidirectional model consistently outperforms the single direction model. From these results, we can conclude that the unlabeled data is beneficial to learn useful latent variables that can be used to decode the corresponding word.

Examining the linguistic characteristics of the models in which our model performs well provides even more interesting insights. Cotterell et al. (2016) estimate how often the inflection process involves prefix changes, stem-internal changes or suffix changes, the results of which are shown in Tab. 2. Among the many languages, the inflection processes of Arabic, Maltese and Navajo are relatively diverse, and contain a large amount of all three forms of inflection. By examining the experimental results together with the morphological inflection process of different languages, we found that among all the languages, Navajo, Maltese and Arabic obtain the largest gains in performance compared with the ensem-
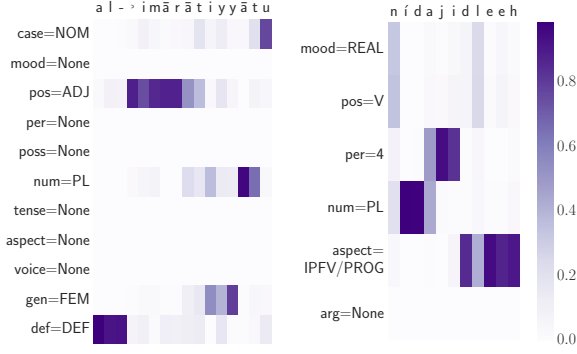
Figure 5: Two examples of attention weights on target linguistic labels: Arabic (**Left**) and Navajo (**Right**). When a tag equals None, it means the word does not have this tag.

Figure 6: Visualization of latent variables **z** for Maltese with 35 pseudo-lemma groups in the figure.

bled MED system. To demonstrate this visually, in Fig. 4, we compare the semi-supervised MSVED with the MED single model w.r.t. the percentage of suffixing inflection of each language, showing this clear trend.

This strongly demonstrates that our model is agnostic to different morphological inflection forms whereas the conventional encoder-decoder with attention on the source input tends to perform better on suffixing-oriented morphological inflection. We hypothesize that for languages that the inflection mostly comes from suffixing, transduction is relatively easy because the source and target words share the same prefix and the decoder can copy the prefix of the source word via attention. However, for languages in which different inflections of a lemma go through different morphological processes, the inflected word and the target word may differ greatly and thus it is crucial to first analyze the lemma of the inflected word before generating the corresponding the reinflection form based on the target labels. This is precisely what our model does by extracting the lemma representation **z** learned by the variational inference model.

6.3 Analysis on Tag Attention

To analyze how the decoder attends to the linguistic labels associated with the target word, we randomly pick two words from the Arabic and Navajo test set and plot the attention weight in Fig. 5. The Arabic word "al-'imārātiyyātu" is an adjective which means "Emirati", and its source word in the test data is "'imārātiyyin" [7]. Both of these are declensions of "'imārātiyy". The source word is
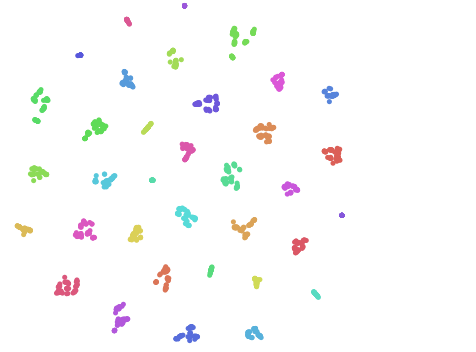
---

[7] https://en.wiktionary.org/wiki/%D8%A5%D9%85%D8%A7%D8%B1%D8%A7%D8%AA%D9%8A

singular, masculine, genitive and indefinite, while the required inflection is plural, feminine, nominative and definite. We can see from the left heat map that the attention weights are turned on at several positions of the word when generating corresponding inflections. For example, "al-" in Arabic is the definite article that marks definite nouns. The same phenomenon can also be observed in the Navajo example, as well as other languages, but due to space limitation, we don't provide detailed analysis here.

6.4 Visualization of Latent Lemmas

To investigate the learned latent representations, in this section we visualize the **z** vectors, examining whether the latent space groups together words with the same lemma. Each sample in SIGMORPHON 2016 contains source word and target words which share the same lemma. We run a heuristic process to assign pairs of words to groups that likely share a lemma by grouping together word pairs for which at least one of the words in each pair shares a surface form. This process is not error free – errors may occur in the case where multiple lemmas share the same surface form – but in general the groupings will generally reflect lemmas except in these rare erroneous cases, so we dub each of these groups a *pseudo-lemma*.

In Fig. 6, we randomly pick 1500 words from Maltese and visualize the continuous latent vectors of these words. We compute the latent vectors as $\mu_\phi(\mathbf{x})$ in the variational posterior inference (Eq. 6) without adding the variance. As expected, words that belong to the same pseudo-lemma (in the same color) are projected into adjacent points in the two-dimensional space. This demonstrates that the continuous latent variable captures the canonical form of a set of words and demonstrates the effectiveness of the proposed representation.

Language	Src Word	Tgt Labels	Gold Tgt	MED	Ours
Turkish	kocama yaratmamdan bitimizde	pos=N,poss=PSS1S,case=ESS,num=SG pos=N,case=NOM,num=SG pos=N,tense=PST,per=1,num=SG	kocamda yaratma bittik	kocama yaratma bitiydik	kocamda yaratman bittim
Maltese	ndammhomli tqożżhieli tissikkmuhomli	pos=V,polar=NEG,tense=PST,num=SG pos=V,polar=NEG,tense=PST,num=SG pos=V,polar=POS,tense=PST,num=PL	tindammhiex tqożżx ssikkmulna	ndammejthiex tqożżx tissikkmulna	tindammhiex qażżejtx tissikkmulna
Finnish	verovapaatta turrumme sukunimin	pos=ADJ,case=PRT,num=PL pos=V,mood=POT,tense=PRS,num=PL pos=N,case=PRIV,num=PL	verovapaita turtunemme sukunimitt	verovappaita turtunemme sukunimeitta	verovapaita turrunemme sukunimeitta

Table 3: Randomly picked output examples on the test data. Within each block, the first, second and third lines are outputs that ours is correct and MED's is wrong, ours is wrong and MED's is correct, both are wrong respectively.

6.5 Analyzing Effects of Size of Unlabeled Data

From Tab. 1, we can see that semi-supervised learning always performs better than supervised learning without unlabeled data. In this section, we investigate to what extent the size of unlabeled data can help with performance. We process a German corpus from a 2017 Wikipedia dump and obtain more than 100,000 German words. These words are ranked in order of occurrence frequency in Wikipedia. The data contains a certain amount of noise since we did not apply any special processing. We shuffle all unlabeled data from both the Wikipedia and the data provided in the shared task used in previous experiments, and increase the number of unlabeled words used in learning by 10,000 each time, and finally use all the unlabeled data (more than 150,000 words) to train the model. Fig. 7 shows that the performance on the test data improves as the amount of unlabeled data increases, which implies that the unsupervised learning continues to help improve the model's ability to model the latent lemma representation even as we scale to a noisy, real, and relatively large-scale dataset. Note that the growth rate of the performance grows slower as more data is added, because although the number of unlabeled data is increasing, the model has seen most word patterns in a relatively small vocabulary.

6.6 Case Study on Reinflected Words

In Tab. 3, we examine some model outputs on the test data from the MED system and our model respectively. It can be seen that most errors of MED and our models can be ascribed to either over-copy or under-copy of characters. In particular, from the complete outputs we observe that our model tends to be more aggressive in its changes, resulting in
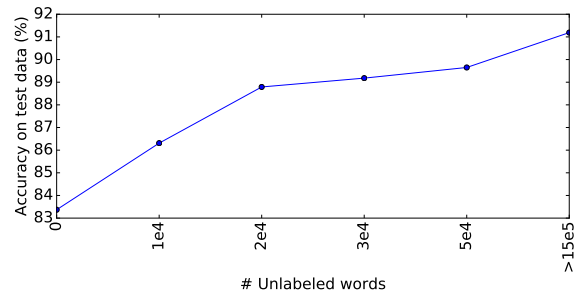

Figure 7: Performance on the German test data w.r.t. the amount of unlabeled Wikipedia data.

it performing more complicated transformations, both successfully (such as Maltese "ndammhomli" to "tindammhiex") and unsuccessfully ("tqożżx" to "qażżejtx"). In contrast, the attentional encoder-decoder model is more conservative in its changes, likely because it is less effective in learning an abstracted representation for the lemma, and instead copies characters directly from the input.

7 Conclusion and Future Work

In this work, we propose a multi-space variational encoder-decoder framework for labeled sequence transduction problem. The MSVED performs well in the task of morphological reinflection, outperforming the state of the art, and further improving with the addition of external unlabeled data. Future work will adapt this framework to other sequence transduction scenarios such as machine translation, dialogue generation, question answering, where continuous and discrete latent variables can be abstracted to guide sequence generation.

Acknowledgments

References

Iñaki Alegria and Izaskun Etxeberria. 2016. Ehu at the sigmorphon 2016 shared task. a simple proposal: Grapheme-to-phoneme for inflection. *In Proceedings of the 2016 Meeting of SIGMORPHON* .

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *The International Conference on Learning Representations* .

Justin Bayer and Christian Osendorfer. 2014. Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610* .

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. *Proceedings of CoNLL* .

Victor Chahuneau, Eva Schlinger, Noah A Smith, and Chris Dyer. 2013. Translating into morphologically rich languages with synthetic phrases. Association for Computational Linguistics.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*. pages 2980–2988.

R. Cotterell, C. Kirov, J. Sylak-Glassman, D. Yarowsky, J. Eisner, and M. Hulden. 2016. The sigmorphon 2016 shared taskmorphological reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Kareem Darwish and Douglas W Oard. 2007. Adapting morphology for arabic information retrieval. In *Arabic Computational Morphology*, Springer, pages 245–262.

Carl Doersch. 2016. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908* .

Greg Durrett and John DeNero. 2013. Supervised learning of complete morphological paradigms. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Atlanta, Georgia, pages 1185–1195.

Otto Fabius and Joost R van Amersfoort. 2014. Variational recurrent auto-encoders. *arXiv preprint arXiv:1412.6581* .

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological inflection generation using character sequence to sequence learning. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, pages 634–643.

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

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2016. Google's multilingual neural machine translation system: Enabling zero-shot translation. *arXiv preprint arXiv:1611.04558* .

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural multi-source morphological reinflection. *arXiv preprint arXiv:1612.06027* .

Katharina Kann and Hinrich Schütze. 2016a. Med: The lmu system for the sigmorphon 2016 shared task on morphological reinflection. In *In Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. Berlin, Germany.

Katharina Kann and Hinrich Schütze. 2016b. Single-model encoder-decoder with explicit morphological representation for reinflection. In *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1328–1338.

Diederik 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*. Montréal, Canada, pages 3581–3589.

D.P. Kingma and M. Welling. 2014. Auto-encoding variational bayes. In *The International Conference on Learning Representations*.

Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. 2016. Semantic parsing with semi-supervised sequential autoencoders. *the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)* .

Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. 2016. Auxiliary deep generative models. *Proceedings of the 33rd International Conference on Machine Learning* .

Chris J Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Advances in Neural Information Processing Systems*. pages 3086–3094.

Yishu Miao and Phil Blunsom. 2016. Language as a latent variable: Discrete generative models for sentence compression. *the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)* .

Garrett Nicolai, Bradley Hauer, Adam St. Arnaud, and Grzegorz Kondrak. 2016. Morphological reinflection via discriminative string transduction. *In Proceedings of the 2016 Meeting of SIGMORPHON* .

Robert Ostling. 2016. Morphological reinflection with convolutional neural networks. *In Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology* page 23.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of The North American Chapter of the Association for Computational Linguistics (NAACL)*. pages 35–40.

Dima Taji, Ramy Eskander, Nizar Habash, and Owen Rambow. 2016. The columbia university - new york university abu dhabi sigmorphon 2016 morphological reinflection shared task submission. *In Proceedings of the 2016 Meeting of SIGMORPHON* .

Keiichi Tokuda, Takashi Masuko, Noboru Miyazaki, and Takao Kobayashi. 2002. Multi-space probability distribution hmm. *IEICE TRANSACTIONS on Information and Systems* 85(3):455–464.

Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. 2008. Applying morphology generation models to machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*. pages 514–522.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* .