Investigating the Impact of Normalizing Flows on Latent Variable Machine Translation

Michael Przystupa [1] Mark Schmidt [1] Muhammad Abdul-Mageed [1]

Abstract

Incorporating latent variables in neural machine translation systems allows explicit representations for lexical and semantic information. These representations help improve general translation quality, as well as provide more robust longer sentence and out-of-domain translations. Previous work has focused on using variational inference with isotropic Gaussian distributions, which we hypothesize cannot sufficiently encode latent factors of language which could exhibit multi-modal distributive behavior. Normalizing flows are an approach that enable more flexible posterior distribution estimates by introduce a change of variables with invertible functions. They have previously been applied successfully in computer vision to enable more flexible posterior distributions of image data. In this work, we present our preliminary results for the effects normalizing flows can have on existing latent variable neural machine translation models as a means to improve translation quality.

1. Introduction

Incorporating latent variables to explicitly capture aspects of language, such as semantics, have previously been shown to improve neural machine translation (NMT) quality. This includes difficult scenarios in machine translation, such as translating longer sentences better (Zhang et al., 2016; Shah & Barber, 2018; Su et al., 2018), demonstrating robustness to domain mis-match between training and test data (Eikema & Aziz, 2018), as well as enabling word level imputation for noisy sentences (Shah & Barber, 2018).

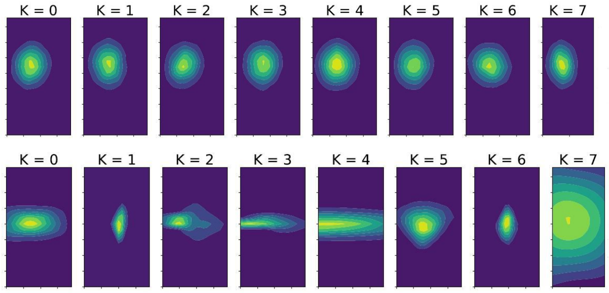

Figure 1. Kernel density estimate contour plots of 10,000 samples from $q(z \mid x, y)$ for each intermittent normalizing flow transformation of the distribution using planar (top) and IAF (bottom) flows. The sentence pair is “Als ich in meinen 20ern war, hatte ich meine erste Psychotherapie-Patientin.” (De), translated to “When I was in my 20s, I saw my very first psychotherapy client.” (En)

Another utility of latent variable NMT systems is encoding lexical variation. This is achieved by sampling from the latent variables and using beam search to find semantically similar sentences (Schulz et al., 2018; Shen et al., 2019). This ability to sample latent variables to produce translations is a valuable feature, because there has been extensive research showing synthetically generate bi-text can lead to improved translation system quality (Sennrich et al., 2015a; Edunov et al., 2018). Depending on the model formulation, latent variable NMT systems can likely help build even better machine translation systems by generating quality synthetic bi-text.

To our knowledge, much of the research in latent variable neural machine translation (LVNMT) applies amortised variational inference to learn the posterior distribution of paired language data. Authors generally have focused on creating variational auto-encoder type models which optimize the evidence lower bound (ELBO) (Kingma & Welling, 2014; Rezende et al., 2014). In the context of translation, this involves maximizing the log-likelihood of the conditional distribution $p(\mathbf{y}|\mathbf{x}, \mathbf{z})$ where $\mathbf{y}$ is the target language, $\mathbf{x}$ is the source language, and $\mathbf{z}$ is the introduced latent variable. Authors have assumed the variational posterior distribution is an isotropic Gaussian and learn a variational distribution $q_\phi(\mathbf{z} \mid \cdot)$ conditioned on different combinations of available

[1] Department of Computer Science, University of British Columbia, British Columbia, Canada. Correspondence to: Michael Przystupa <michael.przystupa@gmail.com>, Mark Schmidt <schmidtm@cs.ubc.ca>, Muhammad Abdul-Mageed <muhammad.mageed@ubc.ca>.

First workshop on *Invertible Neural Networks and Normalizing Flows* (ICML 2019), Long Beach, CA, USA

paired sentences.[1]

One problem with this approach, which is the primary focus of this work, is the choice of variational distribution used to encode information about translation data. A criticism of variational inference is the limited guarantees on approximating, even asymptotically, the true posterior distribution. Particularly in language, there are several empirical findings which suggest that choosing the isotropic Gaussian as the variational distribution family may not truly represent latent aspects of language. One simple example is the power-law distribution behavior that words exhibit in large corpora of text (Koehn, 2010). Even at the character level, previous work in language modeling showed experimental results that exhibit multi-modal distributive behavior (Ziegler & Rush, 2019). These results would suggest that assuming the latent factors follow an isotropic Gaussian distribution is not representative of the true distributive behavior of languages. If latent variables are to be more effectively utilized for machine translation, one needs to consider more flexible variational distributions.

Normalizing flows represent one variational inference approach towards producing more accurate posterior distribution estimates. They accomplish this by transforming a base distribution into a more complex, possibly multi-modal, distribution. This change of variables is achieved by using invertible functions to transform samples from a chosen base distribution (Rezende & Mohamed, 2015). Normalizing flows have been shown to be helpful in computer vision for improving image generation (Kingma et al., 2016; Tomczak & Welling, 2016; Kingma & Dhariwal, 2018; van den Berg et al., 2018), and Schulz et al. (2018) proposed normalizing flows as a potential improvement to their work in LVNMT systems. This variational approach has the added benefit of empirical findings showing more accurate approximations of target posterior distributions when such distributions are known.

We conjecture that normalizing flows are capable of helping achieve better posterior approximations of language factors, and that these improved estimates can help improve the expressiveness of latent codes in machine translation. Figure 1. shows kernel density estimate contour plots of samples after applying normalizing flow transformation of our distribution, as we further explain in Section 5.3.

Overall, we make the following contributions:

1. We discuss the challenges of incorporating normalizing flows to provide benefit for LVNMT systems.
2. We present preliminary experimental findings on German to English translation including results on varying sentence length.
3. We visualize the learned posterior distribution of paired sentences using a 2D latent space to see how the normalizing flows transform the base distribution.

The rest of this paper is organized as follows. In section 2 we discuss related research. In section 3, we review previous works in latent variable machine translation and, as part of our work, we implement one such approach as a probabilistic program. For details on probabilistic programming refer to van de Meent et al. (2018).[2] In section 4, we discuss incorporating normalizing flows into latent variable machine translations systems and the challenges including normalizing flows. Section 5 discusses our experimental results, and we conclude in section 6.

2. Related Work

To our knowledge, the applications of normalizing flows have been considered sparsely in natural language processing, and largely focused on language modeling. Bowman et al. (2015) briefly mention normalizing flows in the context of variational auto-regressive language modeling, but did not report their empirical findings. Ziegler & Rush (2019) provides empirical evidence on applying normalizing flows for character level language modeling. They proposed several non-autoregressive models that allow for parallel decoding, and provided evidence on the multi-modal behaviors of language factors. In this work, we focus on autoregressive approaches applied to neural machine translations.

3. Latent Variable Neural Machine Translation

In this section we give background information on the core aspects of the most successful approaches to neural machine translation, and ways latent variables have been incorporated into existing NMT systems.

3.1. Neural Machine Translation

The leading approaches in NMT are autoregressive sequence to sequence models which include an encoder, decoder, and a generator neural network. For discussion on improvements to this architecture refer to Bahdanau et al. (2014), or Koehn, 2017, chapter 13.5 , and for discussion on alternative architecture choices refer to Koehn, 2017, chapter 13.7.

The encoder is typically a recurrent neural network (RNN), which takes a source word embedding and previous hidden state h_{i-1} as inputs.

[1] Some condition on both the target and source sentence, others on the source, or even just the target sentences

[2] We do not discuss probabilistic programming in this paper, because our work is an application of such languages instead of contributions towards improving probabilistic programming.

$$h_i = \text{RNN}(\text{embed}(x_i), h_{i-1}), \forall i \in d \quad (1)$$

The encoder reads the sequence in order to produce hidden states h_i for each word x_i in a sequence of length d. These states can serve different purposes depending on any additional NMT design choices considered. The most immediate utility is initializing the hidden state of the decoder, which is also an RNN, s_0 with the final encoder hidden state h_d. The decoder follows a similar procedure in equation 1, except the inputs are embed(y_{j-1}) and s_{j-1} instead, for a sequence of length p.

To decode the target words in a sequence, a generator neural network is used where the output is the size of the target language vocabulary. The generator takes at least the decoder hidden states as input, although additional inputs can be included.

$$p(y_j \mid \mathbf{x}, y_{<j}) = \text{generator}(s_j, ...) \quad (2)$$

This function represents the conditional probability of each target word y_j conditioned on the entire source sequence x and previous words $y_{<j}$.

3.2. Latent Variable Neural Machine Translation

Authors have come up with a number of probabilistic formulations to incorporate latent variables into NMT systems. One such approach has focused on modeling the discriminative distribution $p(\mathbf{y}|\mathbf{x})$ which is the typical modeling assumption in regular NMT systems (Zhang et al., 2016).

$$p(\mathbf{y}|\mathbf{x}) = \int_z p(\mathbf{y}, z \mid \mathbf{x}) p(z \mid \mathbf{x}) dz \quad (3)$$

A number of alternatives have also been considered, including latent variables at each time step (Schulz et al., 2018; Su et al., 2018), or instead modeling the joint distribution $p(\mathbf{x}, \mathbf{y})$ (Eikema & Aziz, 2018; Shah & Barber, 2018). For this work, we consider only the discriminative formulation presented by Zhang et al. (2016) and leave expanding to alternative LVNMT formulations as future work.

3.3. Variational Distribution Choice

As can be seen in equation 3, regardless of the model formulation, each model must marginalize out the latent variable $\mathbf{z}$. This is often intractable, and instead previous works have focused on incorporating a variational distribution $q_\phi(\mathbf{z} \mid \cdot)$ in order to approximate the true posterior distribution. There are different ways to condition this distribution, depending whether there is a global latent variable or latent variables-over-time. Here, we discuss the case where the distribution is chosen to be $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})$ and is interpreted as the global semantic latent variable of both source $\mathbf{x}$ and target $\mathbf{y}$ languages.

The challenge with this formulation is that it requires having target sentence y during translation.[3] We choose to follow the approach of Zhang et al. (2016) to address this issue by parameterizing the prior distribution and at decoding use the distribution $p_\theta(\mathbf{z} \mid \mathbf{x})$ instead. The motivation is that both $q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y})$ and $p_\theta(\mathbf{z} \mid \mathbf{x})$ were optimized to match each other such that semantic encoding is preserved in the parameterized prior distribution.

3.4. Encoding Sentences for Latent Distribution

The last consideration is how to actually encode a sentence to generate the distribution parameters. Following other works (Eikema & Aziz, 2018; Shah & Barber, 2018; Zhang et al., 2016), we first encode both the source $\mathbf{x}$ and target $\mathbf{y}$ sentences with an RNN encoder, and produce a $\mathbb{R}^n$ vector, where n is the dimensions of the encoder hidden states. We then apply a mean pooling operation over each collection of hidden states for $h^x_{1:d}$ and $h^y_{1:p}$.

$$\bar{h} = \frac{1}{L} \sum_{i=1}^{L} h_i \quad (4)$$

This produces two vectors $\bar{h}^x$ and $\bar{h}^y$ which are concatenated together $z_0 = [\bar{h}^x; \bar{h}^y]$. z_0 is used to generate the μ_ϕ and σ_ϕ of the variational distribution

$$\mu_\phi = W^{\mu_\phi} z_0 + b^{\mu_\phi}, \sigma_\phi = W^{\sigma_\phi} z_0 + b^{\sigma_\phi} \quad (5)$$

Here W^{μ_ϕ}, b^{μ_ϕ}, W^{σ_ϕ}, and b^{σ_ϕ} are learnable parameters.[4] We use these parameters μ_ϕ and σ_ϕ during training to sample latent code for a sentence with the reparameterization trick (Kingma & Welling, 2014; Rezende et al., 2014) and this latent codes $\mathbf{z}$ is used as input to the decoder RNN$(s_{j-1}, \text{embed}(y_j), \mathbf{z})$.

As previously mentioned, our prior distribution $p_\theta(\mathbf{z} \mid \mathbf{x})$ is also parameterized. A similar approach as the one described for the variational distribution is applied to generate the prior distribution parameters. The key difference is that we instead condition only on $\bar{h}^x$ to generate the distribution parameters μ_θ and σ_θ and separate learnable parameters W^{μ_θ}, b^{μ_θ}, W^{σ_θ}, and b^{σ_θ} are introduced into the model.

[3] Remember that generating $\mathbf{y}$ is the objective of translation.

[4] These parameters could also be expanded to include additional layers and form a fully-connected neural network instead.

4. Normalizing Flows for Machine Translation

In this section we discuss incorporating normalizing flows into latent variable neural machine translation. We also discuss some challenges with incorporating normalizing flows based on the effect they have on the ELBO.

4.1. Applying Flows to Latent Variables

It is relatively easy to incorporate normalizing flows into existing LVNMT models. During the training procedure, one only needs to apply k functions f_i sequentially to samples from the base distribution $p(z_0)$:

$$z_k = f_k \circ f_{k-1} ... \circ f_2 \circ f_1(z_0), z_0 \sim p(z_0) \tag{6}$$

Here, $\circ$ is a shorthand for nested calls of the functions $(f_2(f_1(f_0(z_0))))$. For our experiments, the base distribution $p(z_0)$ refers to our variational distribution $q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y})$ and at decode time we use $p_\theta(\mathbf{z} \mid \mathbf{x})$ which is the same approach taken by Zhang et al. (2016). For deterministic decoding, we set $z_0 = \mu_\theta(x)$ where μ_θ is produced by our parameterized prior distribution, and apply normalizing flows on this fixed value instead of samples from $p_\theta(z_0 \mid x)$

4.2. Challenges with Optimization

The one other consideration with the inclusion of normalizing flows is how they change the ELBO. Here we present a formulation of the ELBO specific to machine translation which is based on the derivation from Rezende & Mohamed, 2015, Section 4.2.

$$\begin{aligned} & E_{q(\mathbf{z}_0 \mid \mathbf{x}, \mathbf{y})} \left[\sum_{j=1}^{U} \log p_\theta(y_j \mid \mathbf{z}^{(k)}, \mathbf{x}, y_{<j}) \right] \\ & - KL(q_\phi(\mathbf{z}_{(0)} \mid \mathbf{x}, \mathbf{y}) || p_\theta(\mathbf{z}^{(k)} \mid \mathbf{x})) \\ & + E_{q_\phi(\mathbf{z}_0 \mid \mathbf{x}, \mathbf{y})} \left[\sum_{k=1}^{K} \log \left| \frac{\delta f^{(k)}}{\delta \mathbf{z}^{(k-1)}} \right| \right] \end{aligned} \tag{7}$$

The first term represent maximizing the likelihood of observed sequences i.e. translating data correctly. The other two terms represent the introduced regularization from the latent variable **z** in the model.

The problem with this objective is that the inclusion of this KL divergence can lead to a problem referred to as ”posterior collapse” (He et al., 2019). This refers to the scenario where, in order to maximize the ELBO, the variational distribution parameters, for all the training data, are pushed to more closely match the prior distribution parameters. In the typical case where the prior is the unit Gaussian distribution, this leads to uninformative codes in which case the latent variable provides no additional information to the model. We recommend Chen et al. (2016) or Zhao et al. (2017) which provide more thorough discussions on the subject.

For this work, we address this potential problem with a previously proposed approach referred to as KL-annealing (Bowman et al., 2015; Sø nderby et al., 2016). KL-annealing is the process of annealing the weight associated to the divergence term in the ELBO from 0.0 (no influence) to 1.0 (original weight). We follow previous research by using a linear annealing schedule to update the weight of our regularization terms after each mini-batch update until it reaches 1.0 and the original ELBO objective is optimized for the remaining duration of training.

4.3. Choice of Normalizing Flows

In the normalizing flows literature, the general trend is to find classes of invertible functions that have more computationally efficient determinants of the Jacobian. This has lead to a variety of normalizing flows available to select from which come with different trade-offs between diverse transformations and fast computation. For our experiments we consider planar flows which are discussed by Rezende & Mohamed (2015) and inverse autoregressive flows discussed by Kingma et al. (2016). We leave exploring alternative choices of flows as future work.

5. Preliminary Experiments

In this section we share our preliminary experimental results on the efficacy of normalizing flows for LVNMT systems. We focus mostly on translation quality as this is the primary usage for such systems. We also visualize the latent space to better understand how normalizing flows can affect the posterior distribution for latent variables.

We specify our models as probabilistic programs using Pyro (Bingham et al., 2018) and run our experiments with Pyro's implementations of normalizing flows. As a starting place of our implementation, we build our models from the tutorial code of Bastings (2018). We train multiple LVNMT models based on the approach considered in Zhang et al. (2016) with an increasing number of normalizing flows and greedily decode translations. To evaluate performance, we use the IWSLT 2016 data sets for German → English translation available through the torchtext library.[5] The BLEU score was measured using Sacrebleu (Post, 2018).

We represent each language's vocabulary with 20,000 byte-pair encodings (BPE) using the SentencePiece API. [6] For details on the motivations and description of BPE in the

[5] https://torchtext.readthedocs.io/en/latest/

[6] https://github.com/google/sentencepiece

Table 1. Best performing models across experiments for German → English translation for different choices of number of flows and latent dimension size. We acquire 0.327 above baseline BLEU with latent dimension size = 2 and 0.165 above baseline with latent dimension size = 50. Bolded scores are best results with each latent dimension size.

Flow Type	# of Flows	Latent Dims	BLEU
Baseline	0	2	19.852
Planar	1	2	**20.180**
IAF	8	2	19.993
Baseline	0	50	19.807
Planar	1	50	**19.972**
IAF	32	50	19.667

context of translation, refer to Sennrich et al., 2015b, Section 3.2.

5.1. General Translation

Our first analysis considers just the general quality of translation when varying the number and type of flows. We compare results based on BLEU for each of our trained LVNMT systems with and without normalizing flows. Figure 2 shows the results based on the BLEU score for LVNMT models trained with an increasing number of normalizing flows. This plot shows the performances of models where the latent dimension was set to 50. The reader will notice we include the performance of 2 runs of our baseline. We report the baseline twice to show the discrepancy in performance as consideration for our observations as these are preliminary findings. These baselines are reported as part of each set of flow experiments even though they are samples from the Gaussian base distribution.

One key observation from Figure 2 is that it appears planar flows perform better than IAF flows despite the number of flows. This suggests that the choice of normalizing flows chosen for LVNMT systems is an important decision. One explanation for this discrepancy is likely related to the expressiveness of each of the considered flows. IAFs were proposed as a more flexible, scalable normalizing flow. In contrast, planar flows are relatively simple transformations making local changes in the distribution space. These findings would seem to suggest that the distributions of latent variables in LVNMT systems do not require much transformation from the original base distribution, although it should be noted our best performances did include at least a single flow.

These findings are consistent even with a change of the dimensions of the latent variables. Table 1 shows the best performances across all our experiments for varying the number of latent variables and number of flows. As previously mentioned, it seems normalizing flows can provide small improvements (0.327 BLEU, sizeable for MT) particularly when choosing a much smaller sized latent space. One reason we suspect a smaller latent space helped was because, particularly in the case of Gaussian distributions, the impact of the KL term is connected to the dimensions of the latent vector. By having a smaller latent space, the loss from the regularization terms in the ELBO contributes less to the loss while still providing useful latent information. However, this may also negatively affect the smooth interpolation of samples in the latent space as the reconstruction loss becomes the predominant factor during optimization.

To fully realize the potential of normalizing flows, additional experiments need to be conducted. One reason for this is because of the size of the chosen dataset for our preliminary experiments. Relatively speaking, typical NMT systems require much larger bi-text corpora to achieve state-of-the art performance (Koehn, 2010; 2017). As normalizing flows introduce additional learning parameters, it is possible our normalizing flow LVNMT model has an insufficient amount of data to see more of a performance benefit.

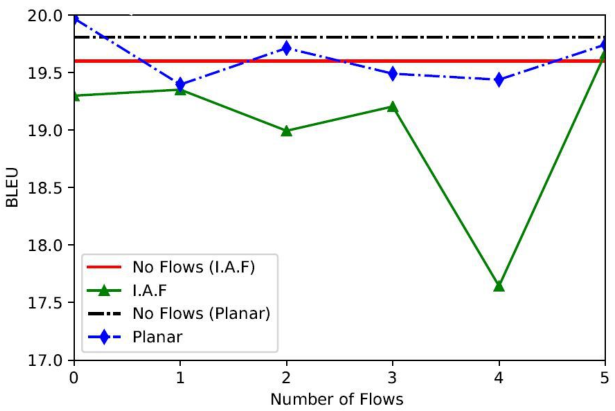

Figure 2. Translation results for German → English with latent dimensions set to 50. The number of flows is on a log scale, actual number of flows are $n = 1, 2, 4, 8, 16, 32$

5.2. Long Sentence Translation

Figure 3 show our results for the BLEU score of sentences of varying length. This plot was again using our models trained with the latent dimensions set to 50. The BLEU score was calculated based on sentence lengths l within ranges of $[l - 4, l]$.

We found mixed results where in some cases it seems normalizing flows on LVNMT models are capable of improving performance on longer sentence translations compared to the baselines. Restricting our comparison to the normalizing flows models, we see that for most sentence lengths, planar flows are more effective than IAF flows. Particularly for sentence lengths of 10 - 40 words, planar flows outperform our

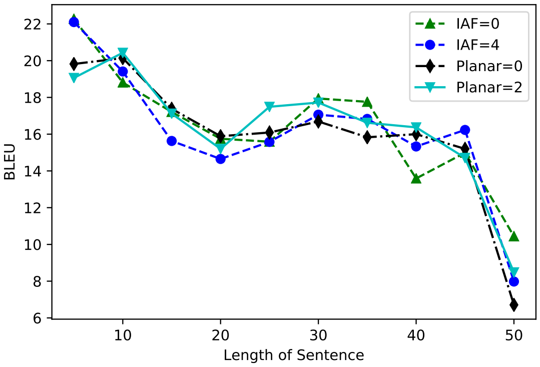

Figure 3. Comparison of BLEU score for sentences of different lengths for best performing models with and without normalizing flows on German → English translation

best IAF flows model as well as for the longest sentences.

However, as we see a discrepancy in performance in the settings with 0 normalizing flows, we report these observations with caution. It is possible these findings could also just be a result of the stochastic behavior in the optimization process by sampling latent variable z. Clearly though we do see that our normalizing flows models do offer comparable performance to the baseline, suggesting further investigation into their utility for longer sentence translation is warranted.

5.3. Visualizing Latent Space

To get a sense of the way the latent space may look like with the addition of normalizing flows, we trained our LVNMT model with the latent dimensions set to 2 in order to visualize the transformations. These models were otherwise trained the same way as the models reported for the general translation systems experiments.

Figure 1 shows our findings for a translation pair from the evaluation set using our LVNMT models with 8 normalizing flows. As we can see, the transformations push around the probability distribution mass, but much of the probability density centers at specific points despite the chosen normalizing flow. This would suggest that the introduced latent variables exhibit uni-modal distributive behavior, contrary to our hypothesis of more multi-modal distributions.

These plots also help illustrate our conclusions in the previous sections on translation quality. The IAF flows transforms the distribution space much more drastically than the planar flows which otherwise only minutely shifts the distribution. Despite the IAF producing more intricate posterior distributions of the latent variable, our results for translation quality would suggest these more intricate distributions are not necessarily helpful to translation quality.

5.4. Discussion

Our results suggest that incorporating normalizing flows into latent variable machine translation systems may provide some improvement. Although the gains we report are small, up to 0.327 BLEU, they are still sizeable for MT. We couch these results with caution, however. By visualizing the latent space, it seems that the information encoded in the latent variable may indeed simply exhibit a uni-modal distribution. Overall, however, we suspect there is potential utility for normalizing flows in machine translation. As we have discussed, larger scale data sets need to be utilized to investigate whether the limitation is available bi-text. We also previously mentioned issues with the ELBO when incorporating normalizing flows and further investigation is needed into it's impact on the loss.

One challenge with a machine translation system is dealing with out of domain data, including needing to translate data from different varieties or dialects within the same language (Zbib et al., 2012). One cited application for latent variables is helping regularize NMT systems to be more robust to unseen data (Eikema & Aziz, 2018). In the future, we intend to evaluate the robustness of our normalizing flow LVNMT models against the baseline LVNMT in the context of out of domain/language variety data.

6. Conclusion

In this paper, we have discussed the general design decisions when specifying latent variable neural machine translation systems. We also presented our preliminary findings when incorporating normalizing flows into such models. Our findings suggest that LVNMT systems can benefit from incorporating normalizing flows, but this improvement depends on the choice of normalizing flow. From these findings, we believe further research on the topic is warranted, and that normalizing flows have potential to help improve future machine translation systems.

Acknowledements

We would like to thank Dr. Frank Wood for helping formulate this idea. We thank Aaron Mischkin who helped us with plotting the variational distribution. We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) and Compute Canada (`www.computecanada.ca`). This work is supported in part by the *Institute for Computing Information and Cognitive Systems (ICICS)* at UBC

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv e-prints*, abs/1409.0473, September 2014. URL https://arxiv.org/abs/1409.0473.

Bastings, J. The annotated encoder-decoder with attention, 2018.

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Horsfall, P., and Goodman, N. D. Pyro: Deep Universal Probabilistic Programming. *Journal of Machine Learning Research*, 2018.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Józefowicz, R., and Bengio, S. Generating sentences from a continuous space. *CoRR*, abs/1511.06349, 2015. URL http://arxiv.org/abs/1511.06349.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. *CoRR*, abs/1611.02731, 2016.

Edunov, S., Ott, M., Auli, M., and Grangier, D. Understanding back-translation at scale. *CoRR*, abs/1808.09381, 2018. URL http://arxiv.org/abs/1808.09381.

Eikema, B. and Aziz, W. Auto-encoding variational neural machine translation. *CoRR*, abs/1807.10564, 2018. URL http://arxiv.org/abs/1807.10564.

He, J., Spokoyny, D., Neubig, G., and Berg-Kirkpatrick, T. Lagging inference networks and posterior collapse in variational autoencoders. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rylDfnCqF7.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 10215–10224. Curran Associates, Inc., 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Kingma, D. P., Salimans, T., and Welling, M. Improving variational inference with inverse autoregressive flow. *CoRR*, abs/1606.04934, 2016.

Koehn, P. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition, 2010. ISBN 0521874157, 9780521874151.

Koehn, P. Neural machine translation. *CoRR*, abs/1709.07809, 2017.

Post, M. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 186–191. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/W18-6319.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/rezende15.html.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In Xing, E. P. and Jebara, T. (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/rezende14.html.

Schulz, P., Aziz, W., and Cohn, T. A stochastic decoder for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1243–1252, Melbourne, Australia, July 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P18-1115.

Sennrich, R., Haddow, B., and Birch, A. Improving neural machine translation models with monolingual data. *CoRR*, abs/1511.06709, 2015a. URL http://arxiv.org/abs/1511.06709.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909, 2015b. URL http://arxiv.org/abs/1508.07909.

Shah, H. and Barber, D. Generative neural machine translation. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 1346–1355. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7409-generative-neural-machine-translation.pdf.

Shen, T., Ott, M., Auli, M., and Ranzato, M. Diverse machine translation with a single multinomial latent variable, 2019. URL https://openreview.net/forum?id=BJgnmhA5KQ.

Sø nderby, C. K., Raiko, T., Maalø e, L., Sø nderby, S. r. K., and Winther, O. Ladder variational autoencoders. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3738–3746. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6275-ladder-variational-autoencoders.pdf.

Su, J., Wu, S., Xiong, D., Lu, Y., Han, X., and Zhang, B. Variational recurrent neural machine translation. *CoRR*, abs/1801.05119, 2018. URL http://arxiv.org/abs/1801.05119.

Tomczak, J. M. and Welling, M. Improving variational autoencoders using householder flow. *CoRR*, abs/1611.09630, 2016.

van de Meent, J., Paige, B., Yang, H., and Wood, F. An introduction to probabilistic programming. *CoRR*, abs/1809.10756, 2018. URL http://arxiv.org/abs/1809.10756.

van den Berg, R., Hasenclever, L., Tomczak, J. M., and Welling, M. Sylvester normalizing flows for variational inference. In *UAI*, 2018.

Zbib, R., Malchiodi, E., Devlin, J., Stallard, D., Matsoukas, S., Schwartz, R., Makhoul, J., Zaidan, O. F., and Callison-Burch, C. Machine translation of arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pp. 49–59, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. ISBN 978-1-937284-20-6. URL http://dl.acm.org/citation.cfm?id=2382029.2382037.

Zhang, B., Xiong, D., and Su, J. Variational neural machine translation. *CoRR*, abs/1605.07869, 2016. URL http://arxiv.org/abs/1605.07869.

Zhao, S., Song, J., and Ermon, S. Infovae: Information maximizing variational autoencoders. *CoRR*, abs/1706.02262, 2017. URL http://arxiv.org/abs/1706.02262.

Ziegler, Z. M. and Rush, A. M. Latent normalizing flows for discrete sequences. *CoRR*, abs/1901.10548, 2019.

A. Supplementary Material

A.1. Experimental Details

We list the hyperparameters and optimization parameters we used in our experiments in table A.1.

Optimization Parameters	
Optimizer	Adam
Learning Rate	0.0003
KL Annealing Schedule	30,000 steps
clip norm	20.0
Mini Batch Size	32
Number of Samples (ELBO)	1
Model Parameters	
Source Embedding Size	300
Target Embedding Size	300
Encoder Hidden Dimensions	256
Number of Encoder Layers	2
Decoder Hidden Dimensions	256
Number of Decoder Layers	2
Dropout	0.5
Z dim (latent variable)	2 or 50
Global Attention Mechanism	
Key Size	512
Query Size	256
I.A.F Details	
Autoregressive NN	150 Units