On the Discrepancy between Density Estimation and Sequence Generation

Jason Lee
New York University
jason@cs.nyu.edu

Dustin Tran
Google AI
trandustin@google.com

Orhan Firat
Google AI
orhanf@google.com

Kyunghyun Cho
New York University
kyunghyun.cho@nyu.edu

Abstract

Many sequence-to-sequence generation tasks, including machine translation and text-to-speech, can be posed as estimating the density of the output y given the input x: $p(y|x)$. Given this interpretation, it is natural to evaluate sequence-to-sequence models using conditional log-likelihood on a test set. However, the goal of sequence-to-sequence generation (or structured prediction) is to find the best output $\hat{y}$ given an input x, and each task has its own downstream metric R that scores a model output by comparing against a set of references y^*: $R(\hat{y}, y^*|x)$. While we hope that a model that excels in density estimation also performs well on the downstream metric, the exact correlation has not been studied for sequence generation tasks. In this paper, by comparing several density estimators on five machine translation tasks, we find that the correlation between rankings of models based on log-likelihood and BLEU varies significantly depending on the range of the model families being compared. First, log-likelihood is highly correlated with BLEU when we consider models within the same family (e.g. autoregressive models, or latent variable models with the same parameterization of the prior). However, we observe no correlation between rankings of models across different families: (1) among non-autoregressive latent variable models, a flexible prior distribution is better at density estimation but gives worse generation quality than a simple prior, and (2) autoregressive models offer the best translation performance overall, while latent variable models with a normalizing flow prior give the highest held-out log-likelihood across all datasets.

1 Introduction

Sequence-to-sequence generation tasks can be cast as conditional density estimation $p(y|x)$ where x and y are input and output sequences. In this framework, density estimators are trained to maximize the conditional log-likelihood, and also evaluated using log-likelihood on a test set. However, many sequence generation tasks require finding the best output $\hat{y}$ given an input x at test time, and the output is evaluated against a set of references y^* on a task-specific metric: $R(\hat{y}, y^*|x)$. For example, machine translation systems are evaluated using BLEU scores (Papineni et al., 2002), image captioning systems use METEOR (Banerjee and Lavie, 2005) and text-to-speech systems use MOS (mean opinion scores). As density estimators are optimized on log-likelihood, we want models with higher held-out log-likelihoods to give better generation quality, but the correlation has not been well studied for sequence generation tasks. In this work, we investigate the correlation between rankings of density estimators based on (1) test log-likelihood and (2) the downstream metric for machine translation.

On five language pairs from three machine translation datasets (WMT'14 En↔De, WMT'16 En↔Ro, IWSLT'16 De→En), we compare the held-out log-likelihood and BLEU scores of several density estimators: (1) autoregressive models (Vaswani et al., 2017), (2) latent variable models with a non-autoregressive decoder and a simple (diagonal Gaussian) prior (Shu et al., 2019), and (3) latent variable models with a non-autoregressive decoder and a flexible (normalizing flow) prior (Ma et al., 2019).

We present two key observations. First, among models within the same family, we find that log-likelihood is strongly correlated with BLEU. The correlation is almost perfect for autoregressive models and high for latent variable models with the same prior. Between models of different families, however, log-likelihood and BLEU are not correlated. Latent variable models with a flow prior are in fact the best density estimators (even better than autoregressive models), but they give the

Proceedings of 4th Workshop on Structured Prediction for NLP, pages 84–94
November 20, 2020. ©2020 Association for Computational Linguistics

worst generation quality. Gaussian prior models offer comparable or better BLEU scores, while autoregressive models give the best BLEU scores overall. From these findings, we conclude that the correlation between log-likelihood and BLEU scores varies significantly depending on the range of model families considered.

Second, we find that knowledge distillation drastically hurts density estimation performance across different models and datasets, but consistently improves translation quality of non-autoregressive models. For autoregressive models, distillation slightly hurts translation quality. Among latent-variable models, iterative inference with a delta posterior (Shu et al., 2019) significantly improves the translation quality of latent variable models with a Gaussian prior, whereas the improvement is relatively small for the flow prior. Overall, for fast generation, we recommend a latent variable non-autoregressive model using a simple prior (rather than a flexible one), knowledge distillation, and iterative inference. This is 5–7x faster than the autoregressive model at the expense of 2 BLEU scores on average, and it improves upon latent variable models with a flexible prior across generation speed, BLEU, and parameter count.

2 Background

Sequence-to-sequence generation is a supervised learning problem of generating an output sequence given an input sequence. For many such tasks, conditional density estimators have been very successful (Sutskever et al., 2014; Bahdanau et al., 2015; Vinyals et al., 2015; Vinyals and Le, 2015).

To learn the distribution of an output sequence, it is crucial to give enough capacity to the model to be able to capture the dependencies among the output variables. We explore two ways to achieve this: (1) directly modeling the dependencies with an autoregressive factorization of the variables, and (2) letting latent variables capture the dependencies, so the distribution of the output sequence can be factorized given the latent variables and therefore more quickly be generated. We discuss both classes of density estimators in depth below. We denote the training set as a set of tuples $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$ and each input and output example as sequences of random variables $\mathbf{x} = \{x_1, \ldots, x_{T'}\}$ and $\mathbf{y} = \{y_1, \ldots, y_T\}$ (where we drop the subscript n for notational simplicity). We use θ to denote the model parameters.

2.1 Autoregressive Models

Learning Autoregressive models factorize the joint distribution of the sequence of output variables $\mathbf{y} = \{y_1, \ldots, y_T\}$ as a product of conditional distributions:

$$\log p_{\text{AR}}(\mathbf{y}|\mathbf{x}) = \sum_{t=1}^{T} \log p_\theta(y_t|y_{<t}, \mathbf{x}).$$

They are trained to maximize the log-likelihood of the training data: $L_{\text{AR}}(\theta) = \frac{1}{N}\sum_{n=1}^{N} \log p_{\text{AR}}(\mathbf{y}_n|\mathbf{x}_n)$.

Parameterization Recurrent neural networks and their gated variants are natural parameterizations of autoregressive models (Elman, 1990; Hochreiter and Schmidhuber, 1997; Chung et al., 2014). By ensuring that no future information $y_{\geq t}$ is used in predicting the current timestep y_t, non-recurrent architectures can also parameterize autoregressive models, such as convolutions (van den Oord et al., 2016; Gehring et al., 2017) and Transformers (Vaswani et al., 2017), which are feedforward networks with self-attention.

Inference Finding the most likely output sequence given an input sequence under an autoregressive model amounts to solving a search problem: $\text{argmax}_{y_{1:T}} \sum_{t=1}^{T} \log p_\theta(y_t|y_{<t}, \mathbf{x})$. As the size of the search space grows exponentially with the length of the output sequence T, solving this exactly is intractable. Therefore, approximate search algorithms are often used such as greedy search or beam search.

2.2 Latent Variable Models

Learning Latent variable models posit a joint distribution of observed variables ($\mathbf{y}$) and unobserved variables ($\mathbf{z}$). They are trained to maximize the marginal log-likelihood of the training data:

$$\log p_{\text{LVM}}(\mathbf{y}|\mathbf{x}) = \log \int_{\mathbf{z}} p_\theta(\mathbf{y}|\mathbf{z}, \mathbf{x})\, p_\theta(\mathbf{z}|\mathbf{x}) d\mathbf{z}. \tag{1}$$

As the marginalization over $\mathbf{z}$ makes computing the marginal log-likelihood and posterior inference intractable, variational inference proposes to use a parameterized family of distributions $q_\phi(\mathbf{z}|\mathbf{y}, \mathbf{x})$ to approximate the true posterior $p(\mathbf{z}|\mathbf{y}, \mathbf{x})$. Then, we have the evidence lowerbound (ELBO) (Wainwright and Jordan, 2008; Kingma and Welling, 2014):

$$\begin{aligned}\log p_{\text{LVM}}(\mathbf{y}|\mathbf{x}) &\geq \text{ELBO}(\mathbf{y},\mathbf{x};\theta,\phi) \qquad (2)\\ &= \mathop{\mathbb{E}}_{\mathbf{z}\sim q_\phi}\left[\log p_\theta(\mathbf{y},\mathbf{z}|\mathbf{x}) - \log q_\phi(\mathbf{z}|\mathbf{y},\mathbf{x})\right],\end{aligned}$$

where $p_\theta(\mathbf{y}|\mathbf{z},\mathbf{x})$ is the decoder, $q_\phi(\mathbf{z}|\mathbf{y},\mathbf{x})$ is the variational posterior and $p_\theta(\mathbf{z}|\mathbf{x})$ is the prior. Both the model and variational parameters θ, ϕ are estimated to maximize ELBO over the training set: $L_{\text{LVM}}(\theta,\phi) = \frac{1}{N}\sum_{n=1}^{N}\text{ELBO}(\mathbf{y}_n,\mathbf{x}_n;\theta,\phi)$.

Parameterization As latent variables can capture the dependencies between the output variables, the decoding distribution can be factorized: $p_\theta(\mathbf{y}|\mathbf{z},\mathbf{x}) = \prod_{t=1}^{T} p_\theta(y_t|\mathbf{z},\mathbf{x})$. The approximate posterior distribution is also often factorized, which can be parameterized by any neural network that outputs mean and standard deviation for each output position: $q_\phi(z_{1:T}|\mathbf{y},\mathbf{x}) = \prod_{t=1}^{T}\mathcal{N}\Big(z_t\Big|\mu_{\phi,t}(\mathbf{y},\mathbf{x}),\sigma_{\phi,t}(\mathbf{y},\mathbf{x})\Big)$. We discuss prior distributions in §2.3.

Inference Generating the most likely output given an input with a latent variable model requires optimizing ELBO with respect to the output: $\text{argmax}_{\mathbf{y}}\text{ELBO}(\mathbf{y},\mathbf{x};\theta,\phi)$. As computing the expectation in Eq. 2 is intractable, we instead optimize a proxy lowerbound using a delta posterior (Shu et al., 2019):

$$\delta(\mathbf{z}|\boldsymbol{\mu}) = \begin{cases}1, & \text{if } \mathbf{z} = \boldsymbol{\mu}\\ 0, & \text{otherwise}\end{cases}$$

Then, the ELBO reduces to:

$$\begin{aligned}&\mathop{\mathbb{E}}_{\mathbf{z}\sim\delta(\mathbf{z}|\boldsymbol{\mu})}\Big[p_\theta(\mathbf{y}|\mathbf{z},\mathbf{x}) + p_\theta(\mathbf{z}|\mathbf{x})\Big] + \overbrace{\mathcal{H}(\delta)}^{=0},\\ &= \log p_\theta(\mathbf{y}|\boldsymbol{\mu},\mathbf{x}) + \log p_\theta(\boldsymbol{\mu}|\mathbf{x}). \qquad (3)\end{aligned}$$

We maximize Eq. 3 with iterative refinement: the EM algorithm alternates between (1) matching the proxy to the original lowerbound by setting $\boldsymbol{\mu} = \mathbb{E}_{q_\phi}[\mathbf{z}]$, and (2) maximizing the proxy lowerbound with respect to $\mathbf{y}$ by: $\hat{\mathbf{y}} = \text{argmax}_{\mathbf{y}}(\log p_\theta(\mathbf{y}|\boldsymbol{\mu},\mathbf{x}))$. The delta posterior is initialized using the prior (e.g. $\boldsymbol{\mu} = \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}[\mathbf{z}]$ in case of a Gaussian prior) so that the inference algorithm is fully deterministic, a desirable property for sequence generation tasks. We study the effect of iterative refinement on BLEU score in detail.

2.3 Prior for Latent Variable Models

Several work have discovered that the prior distribution plays a critical role in balancing the variational posterior and the decoder, and a standard normal distribution may be too rigid for the aggregate posterior to match (Hoffman and Johnson, 2016; Rosca et al., 2018). Indeed, follow-up work found that more flexible prior distributions outperform simple priors on several density estimation tasks (Tomczak and Welling, 2018; Bauer and Mnih, 2019). Therefore, we explore two choices for the prior distribution: a factorized Gaussian and a normalizing flow.

Diagonal Gaussian A simple model of the conditional prior is a factorized Gaussian distribution:

$$\log p_\theta(z_{1:T}|\mathbf{x}) = \sum_{t=1}^{T}\log\mathcal{N}\Big(z_t\Big|\mu_{\theta,t}(\mathbf{x}),\sigma_{\theta,t}(\mathbf{x})\Big),$$

where each latent variable z_t is modeled as a diagonal Gaussian with mean and standard deviation computed from a learned function.

Normalizing Flow Normalizing flows (Tabak and Turner, 2013; Rezende and Mohamed, 2015; Papamakarios et al., 2019) offer a general method to construct complex probability distributions over continuous random variables. It consists of (1) a base distribution $p_b(\epsilon)$ (often chosen as a standard Gaussian distribution) and an invertible transformation f and its inverse f^{-1}, such that $f(\mathbf{z}) = \epsilon,\ f^{-1}(\epsilon) = \mathbf{z}$. As our prior is conditioned on $\mathbf{x}$, so are the transformations: $f(\mathbf{z};\mathbf{x}) = \epsilon,\ f^{-1}(\epsilon;\mathbf{x}) = \mathbf{z}$. Then, by change-of-variables, we can evaluate the exact density of the latent variable $\mathbf{z}$ under the flow prior:

$$\log p_\theta(\mathbf{z}|\mathbf{x}) = \log p_b\Big(f(\mathbf{z};\mathbf{x})\Big) + \log\left|\det\frac{\partial f(\mathbf{z};\mathbf{x})}{\partial\mathbf{z}}\right|.$$

Affine coupling flows (Dinh et al., 2017) enable efficient generation and computation of the Jacobian determinant by constructing each transformation such that only a subset of the random variables undergoes affine transformation, using parameters computed from the remaining variables:

$$\begin{aligned}\mathbf{z}_{\text{id}},\mathbf{z}_{\text{tr}} &= \text{split}(\mathbf{z})\\ \mathbf{s},\mathbf{b} &= g_{\text{param}}(\mathbf{z}_{\text{id}}) \qquad (4)\\ f(\mathbf{z}) &= \text{concat}(\mathbf{z}_{\text{id}};\ \ \mathbf{s}\cdot\mathbf{z}_{\text{tr}} + \mathbf{b}),\end{aligned}$$

where g_{param} can be arbitrarily complex as it needs not be invertible. As invertibility is closed under function composition and the Jacobian determinant is multiplicative, increasingly flexible coupling flows can be constructed by stacking multiple flow layers and reordering such that all the variables are transformed.

2.4 Knowledge Distillation

While most density estimators for sequence generation tasks are trained to maximize the log-likelihood of the training data, recent work have shown that it is possible to improve the performance of non-autoregressive models significantly by training them on the predictions of a pre-trained autoregressive model (Gu et al., 2018; van den Oord et al., 2018). While Zhou et al. (2019) recently found that distillation reduces complexity of the training data, its effect on density estimation performance has not been studied.

3 Problem Definition

On a sequence generation task, a conditional density estimator $F \in \mathcal{H}$ (where $\mathcal{H}$ is a hypothesis set of density estimators in §2) is trained to maximize the log-likelihood (or its approximation) of the training set $\{(x_n, y_n)\}_{n=1}^{N}$:

$$L(F) = \frac{1}{N} \sum_{n=1}^{N} \log p_F(y_n|x_n).$$

Once training converges, the model F is evaluated on the test set $\{(x_m, y_m)\}_{m=1}^{M}$ using a downstream metric R:

$$R(F) = R\big(\{(x_m, y_m, \hat{y}_m)\}_{m=1}^{M}\big),$$

where $\hat{y}_m = \text{argmax}_y \log p_F(y|x_m)$.

To perform model selection, we can rank a set of density estimators $\{F_1, \ldots, F_K\}$ based on either the held-out log-likelihood or the downstream metric. We measure the correlation between the rankings given by the log-likelihood $L(F)$ and the downstream metric $R(F)$.

4 Experimental Setup

On machine translation, we train several autoregressive models and latent variable models and analyze the correlation between their rankings based on log-likelihood and BLEU.

4.1 Datasets and Preprocessing

We use five language pairs from three translation datasets: IWSLT'16 De→En[1] (containing 197K training, 2K development and 2K test sentence pairs), WMT'16 En↔Ro[2] (612K, 2K, 2K pairs) and WMT'14 En↔De[3] (4.5M, 3K, 3K pairs). For WMT'14 En↔De and WMT'16 En↔Ro, both directions are used.

We use the preprocessing scripts with default hyperparameters from the `tensor2tensor` framework.[4] Namely, we use wordpiece tokenization (Schuster and Nakajima, 2012) with 32K wordpieces on all datasets. For WMT'16 En↔Ro, we follow Sennrich et al. (2016) and normalize Romanian and remove diacritics before applying wordpiece tokenization. For training, we discard sentence pairs if either the source or the target length exceeds 64 tokens. As splitting along the time dimension (Ma et al., 2019) in the coupling flow layer requires that the length of the output sequence is a multiple of 2 at each level, `<EOS>` tokens are appended to the target sentence until its length is a multiple of 4.

4.2 Autoregressive Models

We use three Transformer (Vaswani et al., 2017) models of different sizes: Transformer-big (Tr-L), Transformer-base (Tr-B) and Transformer-small (Tr-S). The first two models have the same hyperparameters as in Vaswani et al. (2017). Transformer-small has 2 attention heads, 5 encoder and decoder layers, $d_{\text{model}} = 256$ and $d_{\text{filter}} = 1024$.

4.3 Latent Variable Models

The latent variable models in our experiments are composed of the source sentence encoder, length predictor, prior, decoder and posterior. The source sentence encoder is implemented with a standard Transformer encoder. Given the hidden states of the source sentence, the length predictor (a 2-layer MLP) predicts the length difference between the source and target sentences as a categorical distribution in $[-30, 30]$. We implement the decoder $p_\theta(\mathbf{y}|\mathbf{z}, \mathbf{x})$ with a standard Transformer decoder that outputs the logits of all target tokens in parallel. The approximate posterior $q_\phi(\mathbf{z}|\mathbf{y}, \mathbf{x})$ is implemented as a Transformer decoder with a final Linear layer with weight normalization (Salimans

[1] https://wit3.fbk.eu/

[2] www.statmt.org/wmt16/translation-task.html

[3] www.statmt.org/wmt14/translation-task.html

[4] https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/bin/t2t-datagen

and Kingma, 2016) to output the mean and standard deviation (having dimensionality d_{latent}). Both the decoder and the approximate posterior attend to the source hidden states.

Diagonal Gaussian Prior The diagonal Gaussian prior is implemented with a Transformer decoder which receives a sequence of positional encodings of length T as input, and outputs the mean and standard deviation of each target token (of dimensionality d_{latent}). We train two models of different sizes: Gauss-base (Ga-B) and Gauss-large (Ga-L). Gauss-base has 4 attention heads, 3 posterior layers, 3 decoder layers and 6 encoder layers, whereas Gauss-large has 8 attention heads, 4 posterior layers, 6 decoder layers, 6 encoder layers. $(d_{\text{model}}, d_{\text{latent}}, d_{\text{filter}})$ is (512, 512, 2048) for WMT experiments and (256, 256, 1024) for IWSLT experiments.

Normalizing Flow Prior The flow prior is implemented with Glow (Kingma and Dhariwal, 2018). We use a single Transformer decoder layer with a final Linear layer with weight normalization to parameterize g_{param} in Eq. 4. This produces the shift and scale parameters for the affine transformation. Our flow prior has the multi-scale architecture with three levels (Dinh et al., 2017): at the end of each level, half of the latent variables are modeled with a standard Gaussian distribution. We use three split patterns and multi-headed 1x1 convolution from Ma et al. (2019). We experiment with the following hyperparameter settings: Flow-small (Fl-S) with 12/12/8 flow layers in each level and Flow-base (Fl-B) with 12/24/16 flow layers in each level. The first level corresponds to the latent distribution and the last level corresponds to the base distribution. $(d_{\text{model}}, d_{\text{latent}}, d_{\text{filter}})$ is (320, 320, 640) for all experiments. For the Transformer decoder in g_{param}, we use 4 attention heads for Flow-small and 8 attention heads for Flow-base.

4.4 Training and Optimization

We use the Adam optimizer (Kingma and Ba, 2015) with the learning rate schedule used by Vaswani et al. (2017). The norm of the gradients is clipped at 1.0. We perform early stopping and choose the learning rate warmup steps and dropout rate based on the BLEU score on the development set. To train non-autoregressive models, the loss from the length predictor is minimized jointly with negative ELBO loss.

Knowledge Distillation Following previous work (Kim and Rush, 2016; Gu et al., 2018; Lee et al., 2018), we construct a distilled dataset by decoding the training set using Transformer-base with beam width 4. For IWSLT'16 De→En, we use Transformer-small.

Latent Variable Models To ease optimization of latent variable models (Bowman et al., 2016; Higgins et al., 2017), we set the weight of the KL term to 0 for the first 5,000 SGD steps and linearly increase it to 1 over the next 20,000 steps. Similarly with Mansimov et al. (2019), we find it helpful to add a small regularization term to the training objective that matches the approximate posterior with a standard Gaussian distribution: $\alpha \cdot \text{KL}\big[q_\phi(\mathbf{z}|\mathbf{y}, \mathbf{x}) \,||\, \mathcal{N}(0, \mathbf{I})\big]$, as the original KL term $\text{KL}\big[q_\phi(\mathbf{z}|\mathbf{y}, \mathbf{x}) \,\big|\big|\, p_\theta(\mathbf{z}|\mathbf{x})\big]$ does not have a local point minimum but a valley of minima. We find $\alpha = 10^{-4}$ to work best.

Flow Prior Models We perform data-dependent initialization of actnorm parameters for the flow prior (Kingma and Dhariwal, 2018) at the 5,000-th step, which is at the beginning of KL scheduling.

4.5 Evaluation Metrics

Log-likelihood is the main metric for measuring density estimation (data modeling) performance. We compute exact log-likelihood for autoregressive models. For latent variable models, we estimate the marginal log-likelihood by importance sampling with 1K samples from the approximate posterior and using the ground truth target length.

BLEU measures the similarity (in terms of n-gram overlap) between a generated output and a set of references, regardless of the model. It is a standard metric for generation quality of machine translation systems.

Generation Speed In addition to the quality-driven metrics, we measure the generation speed of each model in the number of sentences generated per second on a single V100 GPU.

5 Results

5.1 Correlation between rankings of models

Table 1 presents the comparison of three model families (Transformer, Gauss, Flow) on five language pairs in terms of generation quality (BLEU) and log-likelihood (LL). We present two sets of results: one from models trained on raw data (Raw),

		BLEU (↑)		LL (↑)	
		Raw	Dist.	Raw	Dist.
WMT'14 En→De	Tr-S	24.54	24.94	-1.77	-2.36
	Tr-B	28.18	27.86	-1.44	-2.19
	Tr-L	29.39	28.29	-1.35	-2.23
	Ga-B	15.74	24.54	-1.51	-2.44
	Ga-L	17.33	**25.53**	-1.47	-2.24
	Fl-S	18.17	21.98	-1.41	-2.13
	Fl-B	18.57	21.82	**-1.23**	-2.05
	Fl-B$^{(*)}$	18.55	21.45		
	Fl-L$^{(*)}$	20.85	23.72		
WMT'14 De→En	Tr-S	29.15	28.40	-1.66	-2.24
	Tr-B	32.21	32.24	-1.42	-2.12
	Tr-L	33.16	32.24	-1.35	-2.05
	Ga-B	21.64	29.29	-1.41	-2.17
	Ga-L	23.03	**30.30**	-1.31	-2.04
	Fl-S	23.17	27.14	-1.28	-1.73
	Fl-B	23.12	26.72	**-1.20**	-1.71
	Fl-B$^{(*)}$	23.36	26.16		
	Fl-L$^{(*)}$	25.40	28.39		
WMT'16 En→Ro	Tr-S	30.12	29.57	-1.72	-1.95
	Tr-B	33.46	33.28	-1.63	-2.52
	Ga-B	28.03	29.71	-2.38	-3.48
	Ga-L	28.16	**30.91**	-2.44	-3.54
	Fl-S	26.85	28.63	-1.53	-2.42
	Fl-B	27.49	29.09	**-1.50**	-2.31
	Fl-B$^{(*)}$	29.26	29.34		
	Fl-L$^{(*)}$	29.86	29.73		
WMT'16 Ro→En	Tr-S	29.33	28.87	-1.84	-1.93
	Tr-B	32.19	31.15	-1.79	-2.28
	Ga-B	26.48	27.81	-2.41	-2.92
	Ga-L	27.35	**28.02**	-2.32	-3.01
	Fl-S	26.03	26.12	-1.65	-2.05
	Fl-B	27.14	27.33	**-1.64**	-2.01
	Fl-B$^{(*)}$	30.16	30.44		
	Fl-L$^{(*)}$	30.69	30.72		
IWSLT	Tr-S	31.54	31.72	-1.84	-2.56
	Ga-B	24.36	26.80	-1.98	-2.70
	Fl-S	23.64	26.69	-1.66	-2.28
	Fl-B	24.89	**27.00**	**-1.57**	-2.46
	Fl-B$^{(*)}$	24.75	27.75		

Table 1: Test BLEU score and log-likelihood of each model. Raw: models trained on raw data. Dist.: models trained on distilled data. Tr-S: Transformer-small. Tr-B: Transformer-base. Tr-L: Transformer-big. Ga-B: Gauss-base. Ga-L: Gauss-large. Fl-S: Flow-small. Fl-B: Flow-base. Fl-L: Flow-large. We use beam search with width 4 for inference with autoregressive models, and one step of iterative inference (Shu et al., 2019) for latent variable models. On most datasets, our Flow-base model gives comparable results to those from Ma et al. (2019), which are denoted with (∗). We boldface the best log-likelihood overall and the best BLEU score among the latent variable models. We underscore best BLEU score among the autoregressive models.

	Tr-B	Ga-B	Fl-B
Raw	0.926	0.831	0.678
Dist.	-0.758	-0.897	-0.873

Table 2: Pearson's correlation between log-likelihood and BLEU across the training checkpoints of Transformer-base, Gauss-base and Flow-base on WMT'14 En→De.

and another from models trained on distilled data (Dist.) (which we mostly discuss in §5.2). We use the original test set in computing the log-likelihood and BLEU scores of the distilled models, so the results are comparable with the undistilled models. We make two main observations:

1. Log-likelihood is highly correlated with BLEU when considering models within the same family.
   (a) Among autoregressive models (Tr-S, Tr-B and Tr-L), there is a perfect correlation between log-likelihood and BLEU. On all five language pairs (undistilled), the rankings of autoregressive models based on log-likelihood and BLEU are identical.
   (b) Among non-autoregressive latent variable models with the same prior distribution, there is a strong but not perfect correlation. Between Gauss-large and Gauss-base, the model with higher held-out log-likelihood also gives higher BLEU on four out of five datasets. Similarly, Flow-base gives higher log-likelihood and BLEU score than Flow-small on all datasets except WMT'14 De→En.
2. Log-likelihood is not correlated with BLEU when comparing models from different families.
   (a) Between latent variable models with different prior distributions, we observe no correlation between log-likelihood and BLEU. On four out of five language pairs (undistilled), Flow-base gives much higher log-likelihood but similar or worse BLEU score than Gauss-base. With distillation, Gauss-large considerably outperforms Flow-base in BLEU on all datasets, while Flow-base gives better log-likelihood.
   (b) Overall, autoregressive models offer the best translation quality but not the best modeling performance. In fact, Flow-base model with a

non-autoregressive decoder gives the highest held-out log-likelihood on all datasets.

Correlation between log-likelihood and BLEU across checkpoints Table 2 presents the correlation between log-likelihood and BLEU across the training checkpoints of several models. The findings are similar to Table 1: for Transformer-base, there is almost perfect correlation (0.926) across the checkpoints. For Gauss-base and Flow-base, we observe strong but not perfect correlation (0.831 and 0.678). Overall, these findings suggest that there is a high correlation between log-likelihood and BLEU when comparing models within the same family. We discuss the correlation for models trained with distillation below in §5.2.

5.2 Knowledge Distillation

In Table 2, we observe a strong negative correlation between log-likelihood and BLEU across the training checkpoints of several density estimators trained with distillation. Indeed, distillation severely hurts density estimation performance on all datasets (see Table 1). In terms of generation quality, it consistently improves non-autoregressive models, yet the amount of improvement varies across models and datasets. On WMT'14 En→De and WMT'14 De→En, distillation gives a significant 7–9 BLEU increase for diagonal Gaussian prior models, but the improvement is relatively smaller on other datasets. Flow prior models benefit less from distillation, only 3–4 BLEU scores on WMT'14 En↔De and less on other datasets. For autoregressive models, distillation results in a slight decrease in generation performance.

5.3 Iterative inference on Gaussian vs. flow prior

We analyze the effect of iterative inference on the Gaussian and the flow prior models. Table 3 shows that iterative refinement improves BLEU and ELBO for both Gaussian prior and flow prior models, but the gain is relatively smaller for the flow prior model.

Visualization of latent space In Figure 1, we visualize the latent space of the approximate prior, the prior and the delta posterior of the latent variable models using t-SNE (van der Maaten, 2014). It is clear from the figures that the delta posterior of Gauss-base has high overlap with the approximate posterior, while the overlap is relatively low for Flow-small. We conjecture that while the loss surface of ELBO contains many local optima that we can reach via iterative refinement, not all of them share the support of the approximate posterior density (hence correspond to data). This is particularly pronounced for the flow prior model.

		NUMBER OF REFINEMENT STEPS			
		0	1	2	4
BLEU	GA-B	22.88	24.36	24.60	24.69
	FL-B	24.57	24.89	24.81	24.92
ELBO	GA-B	-1.11	-0.93	-0.90	-0.89
	FL-B	-1.22	-1.17	-1.16	-1.15

Table 3: Iterative inference with a delta posterior improves BLEU and ELBO for Gauss-base and Flow-base on IWSLT'16 De→En (without distillation).

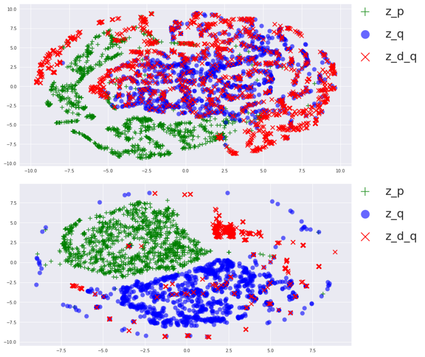

Figure 1: Visualization of the latent space with 1K samples from the prior (green plus sign), the approximate posterior (blue circle) and the delta posterior (red cross) of Gauss-base (top) and Flow-small (bottom) on a IWSLT'16 De→En test example.

5.4 Generation speed and model size

We compare performance, generation speed and size of various models in Table 4. While autoregressive models offer the best translation quality, inference is inherently sequential and slow. Decoding from non-autoregressive latent variable models is much more efficient, and requires constant time with respect to sequence length given parallel computation. Compared to Transformer-base, Gauss-large with 1 step of iterative inference improves generation speed by 6x, at the cost of 2.6 BLEU. On WMT'14 De→En, the performance degradation is 1.9 BLEU. Flow prior models perform much worse than the Gaussian prior models despite having more parameters and slower generation speed.

	BLEU					SPEED					SIZE
$k =$	0	1	2	4	8	0	1	2	4	8	
TR-S	24.54					2.69					17M
TR-B	28.18					2.58					60M
TR-L	29.39					1.93					208M
GA-B	23.15	24.54	24.87	24.94	24.92	28.77	20.52	16.51	12.00	8.11	75M
GA-L	24.31	25.53	25.69	25.68	25.68	19.83	14.72	10.25	7.88	4.91	95M
FL-B	21.57	21.82	21.79	21.81	21.80	5.82	5.60	4.84	3.60	3.37	75M
FL-L$^{(*)}$	23.72										258M

Table 4: BLEU score, generation speed and size of various models on WMT'14 En→De test set. We measure generation speed in sentence/s on a single V100 GPU with batch size 1. We perform inference of autoregressive models using beam search with width 4. For latent variable models, we train perform k steps of iterative inference (Shu et al., 2019) (where $k \in \{0, 1, 2, 4, 8\}$) and report results from models trained with distillation. $(*)$ results are from Ma et al. (2019).

6 Related Work

For sequence generation, the gap between log-likelihood and downstream metric has long been recognized. To address this discrepancy between density estimation and approximate inference (generation), there has largely been two lines of prior work: (1) structured perceptron training for conditional random fields (Lafferty et al., 2001; Collins, 2002; Liang et al., 2006) and (2) empirical risk minimization with approximate inference (Valtchev et al., 1997; Povey and Woodland, 2002; Och, 2003; Qiang Fu and Biing-Hwang Juang, 2007; Stoyanov et al., 2011; Hopkins and May, 2011; Shen et al., 2016). More recent work proposed to train neural sequence models directly on task-specific losses using reinforcement learning (Ranzato et al., 2016; Bahdanau et al., 2017; Jaques et al., 2017) or adversarial training (Goyal et al., 2016).

Despite such a plethora of work in bridging the gap between log-likelihood and the downstream task, the exact correlation between the two has not been established well. Our work investigates the correlation for neural sequence models (autoregressive models and latent variable models) in machine translation. Among autoregressive models for open-domain dialogue, a concurrent work (Adiwardana et al., 2020) found a strong correlation between perplexity and a human evaluation metric that awards sensibleness and specificity. This work confirms a part of our finding that log-likelihood is highly correlated with the downstream metric when we consider models within the same family.

Our work is inspired by recent work on latent variable models for non-autoregressive neural machine translation (Gu et al., 2018; Lee et al., 2018; Kaiser et al., 2018). Specifically, we compare continuous latent variable models with a diagonal Gaussian prior (Shu et al., 2019) and a normalizing flow prior (Ma et al., 2019). We find that while having an expressive prior is beneficial for density estimation, a simple prior delivers better generation quality while being smaller and faster.

7 Conclusion

In this work, we investigate the correlation between log-likelihood and the downstream evaluation metric for machine translation. We train several autoregressive models and latent variable models on five language pairs from three machine translation datasets (WMT'14 En↔De, WMT'16 En↔Ro and IWSLT'16 De→En), and find that the correlation between log-likelihood and BLEU changes drastically depending on the range of model families being compared: Among the models within the same family, log-likelihood is highly correlated with BLEU. Between models of different families, however, we observe no correlation: the flow prior model gives higher held-out log-likelihood but similar or worse BLEU score than the Gaussian prior model. Furthermore, autoregressive models give the highest BLEU scores overall but the latent variable model with a flow prior gives the highest test log-likelihoods on all datasets.

In the future, we will investigate the factors behind this discrepancy. One possibility is the inherent difficulty of inference for latent variable models, which might be resolved by designing better inference algorithms. We will also explore if the discrepancy is mainly caused by the difference in the decoding distribution (autoregressive vs. factorized) or the training objective (maximum likelihood vs. ELBO).

Acknowledgements

We thank our colleagues at the Google Translate and Brain teams, particularly Durk Kingma, Yu Zhang, Yuan Cao and Julia Kreutzer for their feedback on the draft. JL thanks Chunting Zhou, Manoj Kumar and William Chan for helpful discussions.

KC is supported by Samsung Advanced Institute of Technology (Next Generation Deep Learning: from pattern recognition to AI), Samsung Research (Improving Deep Learning using Latent Structure) and NSF Award 1922658 NRT-HDR: FUTURE Foundations, Translation, and Responsibility for Data Science. KC thanks CIFAR, eBay, Naver and NVIDIA for their support.

References

Daniel Adiwardana, Minh-Thang Luong, David R. So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, and Quoc V. Le. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arxiv:2001.09977*.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *5th International Conference on Learning Representations, ICLR*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments.

Matthias Bauer and Andriy Mnih. 2019. Resampled priors for variational autoencoders. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS*, pages 66–75.

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

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arxiv:1412.3555*.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8. Association for Computational Linguistics.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2017. Density estimation using real NVP. In *International Conference on Learning Representations*.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pages 1243–1252.

Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, pages 4601–4609.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR*.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR*.

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

Matthew D Hoffman and Matthew J Johnson. 2016. Elbo surgery: yet another way to carve up the variational evidence lower bound. *Workshop in Advances in Approximate Bayesian Inference, Neurips*.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362. Association for Computational Linguistics.

Natasha Jaques, Shixiang Gu, Dzmitry Bahdanau, José Miguel Hernández-Lobato, Richard E. Turner, and Douglas Eck. 2017. Sequence tutor: Conservative fine-tuning of sequence generation models with kl-control. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pages 1645–1654.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *Proceedings of the*

35th International Conference on Machine Learning, ICML, pages 2395–2404.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*, pages 1317–1327.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*.

Diederik P. Kingma and Prafulla Dhariwal. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems*, pages 10236–10245.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard H. Hovy. 2019. Flowseq: Non-autoregressive conditional sequence generation with generative flow. *arXiv preprint arxiv:1909.02480*.

Laurens van der Maaten. 2014. Accelerating t-sne using tree-based algorithms. *J. Mach. Learn. Res.*, 15(1):3221–3245.

Elman Mansimov, Omar Mahmood, Seokho Kang, and Kyunghyun Cho. 2019. Molecular geometry prediction using a deep generative graph neural network. *arXiv preprint arxiv:1904.00314*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. In *The 9th ISCA Speech Synthesis Workshop*, page 125.

Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. 2018. Parallel wavenet: Fast high-fidelity speech synthesis. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 3915–3923.

George Papamakarios, Eric T. Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2019. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arxiv:1912.02762*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

D. Povey and P. C. Woodland. 2002. Minimum phone error and i-smoothing for improved discriminative training. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–105–I–108.

Qiang Fu and Biing-Hwang Juang. 2007. Automatic speech recognition based on weighted minimum classification error (w-mce) training method. In *2007 IEEE Workshop on Automatic Speech Recognition Understanding (ASRU)*, pages 278–283.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR*.

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

Mihaela Rosca, Balaji Lakshminarayanan, and Shakir Mohamed. 2018. Distribution matching in variational inference. *arXiv preprint arxiv:1802.06847*.

Tim Salimans and Diederik P. Kingma. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29*, page 901.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Edinburgh neural machine translation systems for WMT 16. In *Proceedings of the First Conference on Machine Translation, WMT*, pages 371–376.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2019. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. *arXiv preprint arxiv:1908.07181*.

Veselin Stoyanov, Alexander Ropson, and Jason Eisner. 2011. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS*, pages 725–733.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, pages 3104–3112.

E. G. Tabak and Cristina V. Turner. 2013. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164.

Jakub M. Tomczak and Max Welling. 2018. VAE with a vampprior. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, pages 1214–1223.

V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young. 1997. Mmie training of large vocabulary recognition systems. *Speech Commun.*, 22(4):303–314.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *arXiv preprint arxiv:1506.05869*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3156–3164.

Martin J. Wainwright and Michael I. Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.

Chunting Zhou, Graham Neubig, and Jiatao Gu. 2019. Understanding knowledge distillation in non-autoregressive machine translation. *arXiv preprint arxiv:1911.02727*.