ENGINE: Energy-Based Inference Networks for Non-Autoregressive Machine Translation

Lifu Tu[1] **Richard Yuanzhe Pang**[2*] **Sam Wiseman**[1] **Kevin Gimpel**[1]
[1]Toyota Technological Institute at Chicago, Chicago, IL 60637, USA
[2]New York University, New York, NY 10011, USA
{lifu,swiseman,kgimpel}@ttic.edu, yzpang@nyu.edu

Abstract

We propose to train a non-autoregressive machine translation model to minimize the energy defined by a pretrained autoregressive model. In particular, we view our non-autoregressive translation system as an inference network (Tu and Gimpel, 2018) trained to minimize the autoregressive teacher energy. This contrasts with the popular approach of training a non-autoregressive model on a distilled corpus consisting of the beam-searched outputs of such a teacher model. Our approach, which we call ENGINE (ENerGy-based Inference NEtworks), achieves state-of-the-art non-autoregressive results on the IWSLT 2014 DE-EN and WMT 2016 RO-EN datasets, approaching the performance of autoregressive models.[1]

1 Introduction

The performance of non-autoregressive neural machine translation (NAT) systems, which predict tokens in the target language independently of each other conditioned on the source sentence, has been improving steadily in recent years (Lee et al., 2018; Ghazvininejad et al., 2019; Ma et al., 2019). One common ingredient in getting non-autoregressive systems to perform well is to train them on a corpus of distilled translations (Kim and Rush, 2016). This distilled corpus consists of source sentences paired with the translations produced by a pretrained autoregressive "teacher" system.

As an alternative to training non-autoregressive translation systems on distilled corpora, we instead propose to train them to minimize the *energy* defined by a pretrained autoregressive teacher model. That is, we view non-autoregressive machine translation systems as inference networks (Tu and Gimpel, 2018, 2019; Tu et al., 2019) trained to minimize the teacher's energy. This provides the non-autoregressive model with additional information related to the energy of the teacher, rather than just the approximate minimizers of the teacher's energy appearing in a distilled corpus.

In order to train inference networks to minimize an energy function, the energy must be differentiable with respect to the inference network output. We describe several approaches for relaxing the autoregressive teacher's energy to make it amenable to minimization with an inference network, and compare them empirically. We experiment with two non-autoregressive inference network architectures, one based on bidirectional RNNs and the other based on the transformer model of Ghazvininejad et al. (2019).

In experiments on the IWSLT 2014 DE-EN and WMT 2016 RO-EN datasets, we show that training to minimize the teacher's energy significantly outperforms training with distilled outputs. Our approach, which we call ENGINE (ENerGy-based Inference NEtworks), achieves state-of-the-art results for non-autoregressive translation on these datasets, approaching the results of the autoregressive teachers. Our hope is that ENGINE will enable energy-based models to be applied more broadly for non-autoregressive generation in the future.

2 Related Work

Non-autoregressive neural machine translation began with the work of Gu et al. (2018a), who found benefit from using knowledge distillation (Hinton et al., 2015), and in particular sequence-level distilled outputs (Kim and Rush, 2016). Subsequent work has narrowed the gap between non-autoregressive and autoregressive translation, including multi-iteration refinements (Lee et al.,

[*] Work partly done at Toyota Technological Institute at Chicago and the University of Chicago.

[1] Code is available at https://github.com/lifu-tu/ENGINE

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2819–2826
July 5 - 10, 2020.

2018; Ghazvininejad et al., 2019; Saharia et al., 2020; Kasai et al., 2020) and rescoring with autoregressive models (Kaiser et al., 2018; Wei et al., 2019; Ma et al., 2019; Sun et al., 2019). Ghazvininejad et al. (2020) and Saharia et al. (2020) proposed aligned cross entropy or latent alignment models and achieved the best results of all non-autoregressive models without refinement or rescoring. We propose training inference networks with autoregressive energies and outperform the best purely non-autoregressive methods.

Another related approach trains an "actor" network to manipulate the hidden state of an autoregressive neural MT system (Gu et al., 2017; Chen et al., 2018; Zhou et al., 2020) in order to bias it toward outputs with better BLEU scores. This work modifies the original pretrained network rather than using it to define an energy for training an inference network.

Energy-based models have had limited application in text generation due to the computational challenges involved in learning and inference in extremely large search spaces (Bakhtin et al., 2020). The use of inference networks to output approximate minimizers of a loss function is popular in variational inference (Kingma and Welling, 2013; Rezende et al., 2014), and, more recently, in structured prediction (Tu and Gimpel, 2018, 2019; Tu et al., 2019), including previously for neural MT (Gu et al., 2018b).

3 Energy-Based Inference Networks for Non-Autoregressive NMT

Most neural machine translation (NMT) systems model the conditional distribution $p_\Theta(\boldsymbol{y} \mid \boldsymbol{x})$ of a target sequence $\boldsymbol{y} = \langle y_1, y_2, ..., y_T \rangle$ given a source sequence $\boldsymbol{x} = \langle x_1, x_2, ..., x_{T_s} \rangle$, where each y_t comes from a vocabulary $\mathcal{V}$, y_T is $\langle eos \rangle$, and y_0 is $\langle bos \rangle$. It is common in NMT to define this conditional distribution using an "autoregressive" factorization (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017):

$$\log p_\Theta(\boldsymbol{y} \mid \boldsymbol{x}) = \sum_{t=1}^{|\boldsymbol{y}|} \log p_\Theta(y_t \mid \boldsymbol{y}_{0:t-1}, \boldsymbol{x})$$

This model can be viewed as an energy-based model (LeCun et al., 2006) by defining the **energy function** $E_\Theta(\boldsymbol{x}, \boldsymbol{y}) = -\log p_\Theta(\boldsymbol{y} \mid \boldsymbol{x})$. Given trained parameters Θ, test time inference seeks to find the translation for a given source sentence $\boldsymbol{x}$ with the lowest energy: $\hat{\boldsymbol{y}} = \arg\min_{\boldsymbol{y}} E_\Theta(\boldsymbol{x}, \boldsymbol{y})$.

Finding the translation that minimizes the energy involves combinatorial search. In this paper, we train **inference networks** to perform this search approximately. The idea of this approach is to replace the test time combinatorial search typically employed in structured prediction with the output of a network trained to produce approximately optimal predictions (Tu and Gimpel, 2018, 2019). More formally, we define an inference network $\mathbf{A}_\Psi$ which maps an input $\boldsymbol{x}$ to a translation $\boldsymbol{y}$ and is trained with the goal that $\mathbf{A}_\Psi(\boldsymbol{x}) \approx \arg\min_{\boldsymbol{y}} E_\Theta(\boldsymbol{x}, \boldsymbol{y})$.

Specifically, we train the inference network parameters Ψ as follows (assuming Θ is pretrained and fixed):

$$\widehat{\Psi} = \arg\min_{\Psi} \sum_{\langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathcal{D}} E_\Theta(\boldsymbol{x}, \mathbf{A}_\Psi(\boldsymbol{x})) \quad (1)$$

where $\mathcal{D}$ is a training set of sentence pairs. The network architecture of $\mathbf{A}_\Psi$ can be different from the architectures used in the energy function. In this paper, we combine an autoregressive energy function with a non-autoregressive inference network. By doing so, we seek to combine the effectiveness of the autoregressive energy with the fast inference speed of a non-autoregressive network.

3.1 Energies for Inference Network Training

In order to allow for gradient-based optimization of the inference network parameters Ψ, we now define a more general family of energy functions for NMT. First, we change the representation of the translation $\boldsymbol{y}$ in the energy, redefining $\boldsymbol{y} = \langle \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{|\boldsymbol{y}|} \rangle$ as a sequence of ***distributions*** over words instead of a sequence of words.

In particular, we consider the generalized energy

$$E_\Theta(\boldsymbol{x}, \boldsymbol{y}) = \sum_{t=1}^{|\boldsymbol{y}|} e_t(\boldsymbol{x}, \boldsymbol{y}) \quad (2)$$

where

$$e_t(\boldsymbol{x}, \boldsymbol{y}) = -\boldsymbol{y}_t^\top \log p_\Theta(\cdot \mid \boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}, \boldsymbol{x}). \quad (3)$$

We use the $\cdot$ notation in $p_\Theta(\cdot \mid \ldots)$ above to indicate that we may need the full distribution over words. Note that by replacing the $\boldsymbol{y}_t$ with one-hot distributions we recover the original energy.

In order to train an inference network to minimize this energy, we simply need a network architecture that can produce a sequence of word distributions, which is satisfied by recent non-autoregressive NMT models (Ghazvininejad et al.,

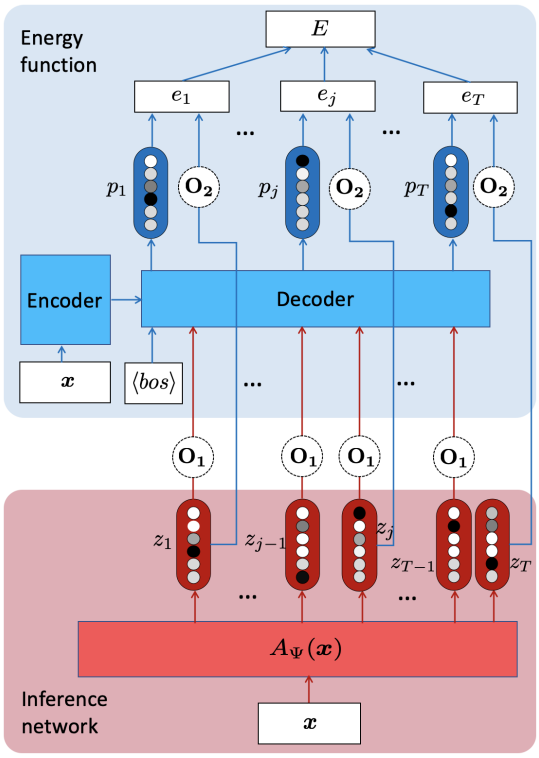

Figure 1: The ENGINE framework trains a non-autoregressive inference network $\mathbf{A}_\Psi$ to produce translations with low energy under a pretrained autoregressive energy E.

2019). However, because the distributions involved in the original energy are one-hot, it may be advantageous for the inference network too to output distributions that are one-hot or approximately so. We will accordingly view inference networks as producing a sequence of T logit vectors $\boldsymbol{z}_t \in \mathbb{R}^{|\mathcal{V}|}$, and we will consider two operators $\mathbf{O_1}$ and $\mathbf{O_2}$ that will be used to map these $\boldsymbol{z}_t$ logits into distributions for use in the energy. Figure 1 provides an overview of our approach, including this generalized energy function, the inference network, and the two operators $\mathbf{O_1}$ and $\mathbf{O_2}$. We describe choices for these operators in the next section.

3.2 Choices for Operators

We now consider ways of defining the two operators that govern the interface between the inference network and the energy function. As shown in Figure 1, we seek an operator $\mathbf{O_1}$ to modulate the way that logits $\boldsymbol{z}_t$ output by the inference network are fed to the decoder input slots in the energy function, and an operator $\mathbf{O_2}$ to determine how the distribution $p_\Theta(\cdot \mid \ldots)$ is used to compute the log probability of a word in $\boldsymbol{y}$. Explicitly, then, we rewrite each local energy term (Eq. 3) as

$$e_t(\boldsymbol{x}, \boldsymbol{y}) = -\mathbf{O_2}(\boldsymbol{z}_t)^\top \log p_\Theta(\cdot \mid \mathbf{O_1}(\boldsymbol{z}_0), \mathbf{O_1}(\boldsymbol{z}_1), \ldots, \mathbf{O_1}(\boldsymbol{z}_{t-1}), \boldsymbol{x}),$$

which our inference networks will minimize with respect to the $\boldsymbol{z}_t$.

The choices we consider for $\mathbf{O_1}$ and $\mathbf{O_2}$, which we present generically for operator $\mathbf{O}$ and logit vector $\boldsymbol{z}$, are shown in Table 1, and described in more detail below. Some of these $\mathbf{O}$ operations are not differentiable, and so the Jacobian matrix $\frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$ must be approximated during learning; we show the approximations we use in Table 1 as well.

We consider five choices for each $\mathbf{O}$:

(a) **SX**: softmax. Here $\mathbf{O}(\boldsymbol{z}) = \mathrm{softmax}(\boldsymbol{z})$; no Jacobian approximation is necessary.

(b) **STL**: straight-through logits. Here $\mathbf{O}(\boldsymbol{z}) = \mathit{onehot}(\arg\max_i \boldsymbol{z})$. $\frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$ is approximated by the identity matrix $\boldsymbol{I}$ (see Bengio et al. (2013)).

(c) **SG**: straight-through Gumbel-Softmax. Here $\mathbf{O}(\boldsymbol{z}) = \mathit{onehot}(\arg\max_i \mathrm{softmax}(\boldsymbol{z} + \boldsymbol{g}))$, where g_i is Gumbel noise.[2] $\frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$ is approximated with $\frac{\partial \mathrm{softmax}(\boldsymbol{z}+\boldsymbol{g})}{\partial \boldsymbol{z}}$ (Jang et al., 2016).

(d) **ST**: straight-through. This setting is identical to SG with $\boldsymbol{g} = \mathbf{0}$ (see Bengio et al. (2013)).

(e) **GX**: Gumbel-Softmax. Here $\mathbf{O}(\boldsymbol{z}) = \mathrm{softmax}(\boldsymbol{z} + \boldsymbol{g})$, where again g_i is Gumbel noise; no Jacobian approximation is necessary.

	$\mathbf{O}(\boldsymbol{z})$	$\frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$
SX	$\boldsymbol{q}$	$\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{z}}$
STL	$\mathit{onehot}(\arg\max(\boldsymbol{z}))$	$\boldsymbol{I}$
SG	$\mathit{onehot}(\arg\max(\tilde{\boldsymbol{q}}))$	$\frac{\partial \tilde{\boldsymbol{q}}}{\partial \tilde{\boldsymbol{z}}}$
ST	$\mathit{onehot}(\arg\max(\boldsymbol{q}))$	$\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{z}}$
GX	$\tilde{\boldsymbol{q}}$	$\frac{\partial \tilde{\boldsymbol{q}}}{\partial \tilde{\boldsymbol{z}}}$

Table 1: Let $\mathbf{O}(\boldsymbol{z}) \in \Delta^{|\mathcal{V}|-1}$ be the result of applying an $\mathbf{O_1}$ or $\mathbf{O_2}$ operation to logits $\boldsymbol{z}$ output by the inference network. Also let $\tilde{\boldsymbol{z}} = \boldsymbol{z} + \boldsymbol{g}$, where $\boldsymbol{g}$ is Gumbel noise, $\boldsymbol{q} = \mathrm{softmax}(\boldsymbol{z})$, and $\tilde{\boldsymbol{q}} = \mathrm{softmax}(\tilde{\boldsymbol{z}})$. We show the Jacobian (approximation) $\frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$ we use when computing $\frac{\partial Loss}{\partial \boldsymbol{z}} = \frac{\partial Loss}{\partial \mathbf{O}(\boldsymbol{z})} \frac{\partial \mathbf{O}(\boldsymbol{z})}{\partial \boldsymbol{z}}$, for each $\mathbf{O}(\boldsymbol{z})$ considered.

[2] $g_i = -\log(-\log(u_i))$ and $u_i \sim \mathbf{Uniform}(0, 1)$.

$\mathbf{O_1} \setminus \mathbf{O_2}$	SX	STL	SG	ST	GX
SX	**55 (20.2)**	256 (0)	56 (19.6)	**55 (20.1)**	55 (19.6)
STL	97 (14.8)	164 (8.2)	94 (13.7)	95 (14.6)	190 (0)
SG	82 (15.2)	206 (0)	81 (14.7)	82 (15.0)	83 (13.5)
ST	81 (14.7)	170 (0)	81 (14.4)	80 (14.3)	83 (13.7)
GX	**53 (19.8)**	201 (0)	56 (18.3)	54 (19.6)	55 (19.4)

(a) seq2seq AR energy, BiLSTM inference networks

SX	STL	SG	ST	GX
80 (31.7)	133 (27.8)	81 (31.5)	**80 (31.7)**	81 (31.6)
186 (25.3)	133 (27.8)	95 (20.0)	97 (30.1)	180 (26.0)
98 (30.1)	133 (27.8)	95 (30.1)	97 (30.0)	97 (29.8)
98 (30.2)	133 (27.8)	95 (30.0)	97 (30.1)	97 (30.0)
81 (31.5)	133 (27.8)	81 (31.2)	81 (31.5)	81 (31.4)

(b) transformer AR energy, CMLM inference networks

Table 2: Comparison of operator choices in terms of energies (BLEU scores) on the IWSLT14 DE-EN dev set with two energy/inference network combinations. Oracle lengths are used for decoding. $\mathbf{O_1}$ is the operation for feeding inference network outputs into the decoder input slots in the energy. $\mathbf{O_2}$ is the operation for computing the energy on the output. Each row corresponds to the same $\mathbf{O_1}$, and each column corresponds to the same $\mathbf{O_2}$.

4 Experimental Setup

4.1 Datasets

We evaluate our methods on two datasets: IWSLT14 German (DE) → English (EN) and WMT16 Romanian (RO) → English (EN). All data are tokenized and then segmented into subword units using byte-pair encoding (Sennrich et al., 2016). We use the data provided by Lee et al. (2018) for RO-EN.

4.2 Autoregressive Energies

We consider two architectures for the pretrained autoregressive (AR) energy function. The first is an autoregressive sequence-to-sequence (seq2seq) model with attention (Luong et al., 2015). The encoder is a two-layer BiLSTM with 512 units in each direction, the decoder is a two-layer LSTM with 768 units, and the word embedding size is 512. The second is an autoregressive transformer model (Vaswani et al., 2017), where both the encoder and decoder have 6 layers, 8 attention heads per layer, model dimension 512, and hidden dimension 2048.

4.3 Inference Network Architectures

We choose two different architectures: a BiLSTM "tagger" (a 2-layer BiLSTM followed by a fully-connected layer) and a conditional masked language model (CMLM; Ghazvininejad et al., 2019), a transformer with 6 layers per stack, 8 attention heads per layer, model dimension 512, and hidden dimension 2048. Both architectures require the target sequence length in advance; methods for handling length are discussed in Sec. 4.5. For baselines, we train these inference network architectures as non-autoregressive models using the standard per-position cross-entropy loss. For faster inference network training, we initialize inference networks with the baselines trained with cross-entropy loss in our experiments.

The baseline CMLMs use the partial masking strategy described by Ghazvininejad et al. (2019). This involves using some masked input tokens and some provided input tokens during training. At test time, multiple iterations ("refinement iterations") can be used for improved results (Ghazvininejad et al., 2019). Each iteration uses partially-masked input from the preceding iteration. We consider the use of multiple refinement iterations for both the CMLM baseline and the CMLM inference network.[3]

4.4 Hyperparameters

For inference network training, the batch size is 1024 tokens. We train with the Adam optimizer (Kingma and Ba, 2015). We tune the learning rate in $\{5e{-}4, 1e{-}4, 5e{-}5, 1e{-}5, 5e{-}6, 1e{-}6\}$. For regularization, we use L2 weight decay with rate 0.01, and dropout with rate 0.1. We train all models for 30 epochs. For the baselines, we train the models with local cross entropy loss and do early stopping based on the BLEU score on the dev set. For the inference network, we train the model to minimize the energy (Eq. 1) and do early stopping based on the energy on the dev set.

4.5 Predicting Target Sequence Lengths

Non-autoregressive models often need a target sequence length in advance (Lee et al., 2018). We report results both with oracle lengths and with a simple method of predicting it. We follow Ghazvininejad et al. (2019) in predicting the length of the

[3]The CMLM inference network is trained according to Eq. 1 with full masking (no partial masking like in the CMLM baseline). However, since the CMLM inference network is initialized using the CMLM baseline, which is trained using partial masking, the CMLM inference network is still compatible with refinement iterations at test time.

	IWSLT14 DE-EN		WMT16 RO-EN	
	# iterations		# iterations	
	1	10	1	10
CMLM	28.11	33.39	28.20	33.31
ENGINE	31.99	33.17	33.16	34.04

Table 3: Test BLEU scores of non-autoregressive models using no refinement (# iterations = 1) and using refinement (# iterations = 10). Note that the # iterations = 1 results are purely non-autoregressive. ENGINE uses a CMLM as the inference network architecture and the transformer AR energy. The length beam size is 5 for CMLM and 3 for ENGINE.

translation using a representation of the source sequence from the encoder. The length loss is added to the cross-entropy loss for the target sequence. During decoding, we select the top $k = 3$ length candidates with the highest probabilities, decode with the different lengths in parallel, and return the translation with the highest average of log probabilities of its tokens.

5 Results

Effect of choices for $\mathbf{O_1}$ and $\mathbf{O_2}$. Table 2 compares various choices for the operations $\mathbf{O_1}$ and $\mathbf{O_2}$. For subsequent experiments, we choose the setting that feeds the whole distribution into the energy function ($\mathbf{O_1}$ = SX) and computes the loss with straight-through ($\mathbf{O_2}$ = ST). Using Gumbel noise in $\mathbf{O_2}$ has only minimal effect, and rarely helps. Using ST instead also speeds up training by avoiding the noise sampling step.

Training with distilled outputs vs. training with energy. We compared training non-autoregressive models using the references, distilled outputs, and as inference networks on both datasets. Table 5 in the Appendix shows the results when using BiLSTM inference networks and seq2seq AR energies. The inference networks improve over training with the references by 11.27 BLEU on DE-EN and 12.22 BLEU on RO-EN. In addition, inference networks consistently improve over non-autoregressive networks trained on the distilled outputs.

Impact of refinement iterations. Ghazvininejad et al. (2019) show improvements with multiple refinement iterations. Table 3 shows refinement results of CMLM and ENGINE. Both improve with multiple iterations, though the improvement is much larger with CMLM. However, even with 10 iterations, ENGINE is comparable to CMLM on DE-EN and outperforms it on RO-EN.

	IWSLT14 DE-EN	WMT16 RO-EN
Autoregressive (Transformer)		
Greedy Decoding	33.00	33.33
Beam Search	34.11	34.07
Non-autoregressive		
Iterative Refinement (Lee et al., 2018)	-	25.73†
NAT with Fertility (Gu et al., 2018a)	-	29.06†
CTC (Libovický and Helcl, 2018)	-	24.71†
FlowSeq (Ma et al., 2019)	27.55†	30.44†
CMLM (Ghazvininejad et al., 2019)	28.25	28.20†
Bag-of-ngrams-based loss (Shao et al., 2020)	-	29.29†
AXE CMLM (Ghazvininejad et al., 2020)	-	31.54†
Imputer-based model (Saharia et al., 2020)	-	31.7†
ENGINE (ours)	**31.99**	**33.16**

Table 4: BLEU scores on two datasets for several non-autoregressive methods. The inference network architecture is the CMLM. For methods that permit multiple refinement iterations (CMLM, AXE CMLM, ENGINE), one decoding iteration is used (meaning the methods are purely non-autoregressive). †Results are from the corresponding papers.

Comparison to other NAT models. Table 4 shows 1-iteration results on two datasets. To the best of our knowledge, ENGINE achieves state-of-the-art NAT performance: 31.99 on IWSLT14 DE-EN and 33.16 on WMT16 RO-EN. In addition, ENGINE achieves comparable performance with the autoregressive NMT model.

6 Conclusion

We proposed a new method to train non-autoregressive neural machine translation systems via minimizing pretrained energy functions with inference networks. In the future, we seek to expand upon energy-based translation using our method.

Acknowledgments

We would like to thank Graham Neubig for helpful discussions and the reviewers for insightful comments. This research was supported in part by an Amazon Research Award to K. Gimpel.

References

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

Anton Bakhtin, Yuntian Deng, Sam Gross, Myle Ott, Marc'Aurelio Ranzato, and Arthur Szlam. 2020. Energy-based models for text.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Yun Chen, Victor O.K. Li, Kyunghyun Cho, and Samuel Bowman. 2018. A stable and effective learning strategy for trainable greedy decoding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 380–390, Brussels, Belgium. Association for Computational Linguistics.

Marjan Ghazvininejad, Vladimir Karpukhin, Luke Zettlemoyer, and Omer Levy. 2020. Aligned cross entropy for non-autoregressive machine translation. *arXiv preprint arXiv:2004.01655*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6111–6120, Hong Kong, China. Association for Computational Linguistics.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018a. Non-autoregressive neural machine translation. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Jiatao Gu, Kyunghyun Cho, and Victor O.K. Li. 2017. Trainable greedy decoding for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1978, Copenhagen, Denmark. Association for Computational Linguistics.

Jiatao Gu, Daniel Jiwoong Im, and Victor O. K. Li. 2018b. Neural machine translation with Gumbel-greedy decoding. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5125–5132. AAAI Press.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. In *Proceedings of International Conference on Learning Representations (ICLR)*.

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 *International Conference on Machine Learning*, pages 2395–2404.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Parallel machine translation with disentangled context transformer.

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

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu-Jie Huang. 2006. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press.

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, Brussels, Belgium. Association for Computational Linguistics.

Jindřich Libovický and Jindřich Helcl. 2018. End-to-end non-autoregressive neural machine translation with connectionist temporal classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3016–3021, Brussels, Belgium. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Xuezhe Ma, Chunting Zhou, Xian Li, Graham Neubig, and Eduard Hovy. 2019. FlowSeq: Non-autoregressive conditional sequence generation with generative flow. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4281–4291, Hong Kong, China. Association for Computational Linguistics.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. *arXiv preprint arXiv:2004.07437*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Chenze Shao, Jinchao Zhang, Yang Feng, Fandong Meng, and Jie Zhou. 2020. Minimizing the bag-of-ngrams difference for non-autoregressive neural machine translation. In *AAAI*.

Zhiqing Sun, Zhuohan Li, Haoqing Wang, Di He, Zi Lin, and Zhihong Deng. 2019. Fast structured decoding for sequence models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3016–3026. Curran Associates, Inc.

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

Lifu Tu and Kevin Gimpel. 2018. Learning approximate inference networks for structured prediction. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Lifu Tu and Kevin Gimpel. 2019. Benchmarking approximate inference methods for neural structured prediction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3313–3324, Minneapolis, Minnesota. Association for Computational Linguistics.

Lifu Tu, Richard Yuanzhe Pang, and Kevin Gimpel. 2019. Improving joint training of inference networks and structured prediction energy networks. *arXiv preprint arXiv:1911.02891*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Bingzhen Wei, Mingxuan Wang, Hao Zhou, Junyang Lin, and Xu Sun. 2019. Imitation learning for non-autoregressive neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1304–1312, Florence, Italy. Association for Computational Linguistics.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2020. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations (ICLR)*.

A Appendix

A.1 Training with Distilled Outputs vs. Training with Energy

In order to compare ENGINE with training on distilled outputs, we train BiLSTM models in three ways: "baseline" which is trained with the human-written reference translations, "distill" which is trained with the distilled outputs (generated using the autoregressive models), and "ENGINE", our method which trains the BiLSTM as an inference network to minimize the pretrained seq2seq autoregressive energy. Oracle lengths are used for decoding. Table 5 shows test results for both datasets, showing significant gains of ENGINE over the baseline and distill methods. Although the results shown here are lower than the transformer results, the trend is clearly indicated.

	IWSLT14 DE-EN		WMT16 RO-EN	
	Energy (↓)	BLEU (↑)	Energy (↓)	BLEU (↑)
baseline	153.54	8.28	175.94	9.47
distill	112.36	14.58	205.71	5.76
ENGINE	51.98	19.55	64.03	21.69

Table 5: Test results of non-autoregressive models when training with the references ("baseline"), distilled outputs ("distill"), and energy ("ENGINE"). Oracle lengths are used for decoding. Here, ENGINE uses BiLSTM inference networks and pretrained seq2seq AR energies. ENGINE outperforms training on both the references and a pseudocorpus.

A.2 Analysis of Translation Results

In Table 6, we present randomly chosen translation outputs from WMT16 RO-EN. For each Romanian sentence, we show the reference from the dataset, the translation from CMLM, and the translation

Source:
seful onu a solicitat din nou tuturor partilor , inclusiv consiliului de securitate onu divizat sa se unifice si sa sustina negocierile pentru a gasi o solutie politica .
Reference :
the u.n. chief again urged all parties , including the divided u.n. security council , to unite and support inclusive negotiations to find a political solution .
CMLM :
the un chief again again urged all parties , including the divided un security council to unify and support negotiations in order to find a political solution .
ENGINE :
the un chief has again urged all parties , including the divided un security council to unify and support negotiations in order to find a political solution .

Source:
adevarul este ca a rupt o racheta atunci cand a pierdut din cauza ca a acuzat crampe in us , insa nu este primul jucator care rupe o racheta din frustrare fata de el insusi si il cunosc pe thanasi suficient de bine incat sa stiu ca nu s @-@ ar mandri cu asta .
Reference :
he did break a racquet when he lost when he cramped in the us , but he 's not the first player to break a racquet out of frustration with himself , and i know thanasi well enough to know he wouldn 't be proud of that .
CMLM :
the truth is that it has broken a rocket when it lost because accused crcrpe in the us , but it is not the first player to break rocket rocket rocket frustration frustration himself himself and i know thanthanasi enough enough know know he would not be proud of that .
ENGINE :
the truth is that it broke a rocket when it lost because he accused crpe in the us , but it is not the first player to break a rocket from frustration with himself and i know thanasi well well enough to know he would not be proud of it .

Source:
realizatorii studiului mai transmit ca " romanii simt nevoie de ceva mai multa aventura in viata lor (24 %) , urmat de afectiune (21 %) , bani (21 %) , siguranta (20 %) , nou (19 %) , sex (19 %) , respect 18 % , incredere 17 % , placere 17 % , conectare 17 % , cunoastere 16 % , protectie 14 % , importanta 14 % , invatare 12 % , libertate 11 % , autocunoastere 10 % si control 7 % " .
Reference :
the study 's conductors transmit that " romanians feel the need for a little more adventure in their lives (24 %) , followed by affection (21 %) , money (21 %) , safety (20 %) , new things (19 %) , sex (19 %) respect 18 % , confidence 17 % , pleasure 17 % , connection 17 % , knowledge 16 % , protection 14 % , importance 14 % , learning 12 % , freedom 11 % , self @-@ awareness 10 % and control 7 % . "
CMLM :
survey survey makers say that ' romanians romanians some something adventadventure ure their lives 24 24 %) followed followed by % % % % % , (21 % %), safety (% % %), new19% %),), 19 % % %), respect 18 % % % % % % % % , , % % % % % % % % , , % , 14 % , 12 % %
ENGINE :
realisation of the survey say that ' romanians feel a slightly more adventure in their lives (24 %) followed by aff% (21 %) , money (21 %), safety (20 %) , new 19 %) , sex (19 %) , respect 18 % , confidence 17 % , 17 % , connecting 17 % , knowledge % % , 14 % , 14 % , 12 % %

Table 6: Examples of translation outputs from ENGINE and CMLM on WMT16 RO-EN without refinement iterations.

from ENGINE. We observe that without the refinement iterations, CMLM performs well for shorter source sentences. However, it still prefers generating repeated tokens. ENGINE, on the other hand, generates much better translations with fewer repeated tokens.