VARIATIONAL NN in NMT
Report 2

NGO HO Anh Khoa

Problem **1**

Solution proposed: Neural Variational Inference **1**

Main idea 1

Detail 1

Advantage 1

Problem 2

Neural Variational Inference **2**

Variational bound and Reparameterization [Kingma et al] 2

Language modeling 3

Neural Variational Inference Framework in [Miao et al] 3

Language model of [Miao et al] 4

Language model of [Bowman et al] 5

Approach for NMT [Zhang et al] 6

Variational Neural Encoder 6

Variational Neural Inferer 6

Variational Neural Decoder 7

GANs and Continuous Space in Global Feature Learning [Donahue et al] **7**

GAN and Variational Training in Energy Based Models [Zhai et al] **9**

Reference **10**

A. Problem

GAN is designed only for the continuous data, which means that it has difficulties in directly generating sequences of discrete tokens. In other words, the gradient of the output of the discriminator shows how to slightly change the data. However, if the output is a word - discrete token, it is impossible to slightly modify this word. There are two approaches: Reinforcement Learning [Report 1] or Continuous Space. This report concentrates on the second approach with Variational Inference.

For NMT, Variational Inference is probably a good approach for long sentence translation because of its semantic representation of entire sentence. In addition, one ability of Variational Inference should be considered is data augmentation.

B. Solution proposed: Neural Variational Inference

a. Main idea

An inference network approximates the intractable distributions over the latent variables. In detail, the discrete text input is transformed into a continuous space. The hidden state is sampled from this space. Therefore, the model concentrates in learning the posterior probability, which make a strong generalisation.

b. Detail

There are two components including an probabilistic encoder and density-based probabilistic decoder. [Kingma et al]

i. An encoder is a recognition model, based on a given sample, generating a distribution over the possible values of the latent variables (Hidden code). The given sample could be generated reversely from the latent variables.
ii. An decoder, based on a given hidden code, produces a distribution over the possible corresponding values of the given sample.

c. Advantage

i. Semantic representation:

For language modeling, [Bowman et al] shows that the ability of this approach is an explicit global distributed sentence representation or a global semantic signal. In NMT, [Zhang et al] shows the improvement for the long sentence translation. In fact, the hidden code represents the global features (Semantic of entire sentence) of a sentence which takes an important role in this kind of translation.

ii. Generalization ability:

[Kingma et al] mentions that the model allow effective generalization from a small labeled dataset to a large unlabeled dataset. Therefore, this approaches underlines the strong ability of generalization. In the case of a small parallel corpus, we could have more source - target sentences. It notes that a strong generalization reduces the problem of overfitting.

Moreover, it simplifies the sampling procedure by defining a differentiable, deterministic mapping from the simple distribution into the input space [Zhai et al]. It avoids the expensive iterative inference schemes MCMC per sample [Kingma et al].

d. Problem

-

C. Neural Variational Inference

[Miao et al] and [Bowman et al], based on the variational auto-encoder of [Kingma et al], proposes two similar approaches for language model.

a. Variational bound and Reparameterization [Kingma et al]

The variational lower bound on the marginal likelihood of a sample x includes a KL-divergence of the posterior from the prior (nonnegative) and a data likelihood under the posterior:

$$L(\theta, \varphi, x) \;=\; -D_{KL}(\, q_\varphi(z|x) \,\|\, p_\theta(z)\,) \;+\; E_{\,q_\varphi(z|x)}[\, log\, p_\theta(x|z)\,]$$

- x: Sample - one datapoint
- z: Hidden code
- $q_\varphi(z|x)$: Encoder and its variational parameters φ
- $p_\theta(x|z)$: Decoder and its generative parameters θ

A good encoder which encodes the useful information in the hidden code z has a non-zero KL-divergence term and a relatively small cross entropy term.

The differentiation and the optimisation of the lower bound φ faces the fact that Monte Carlo gradient estimator, an usual approach, shows a very high variance and impractical estimator. Therefore, the random variable $\overline{z} \sim q_\varphi(z|x)$ is reparameterized by using a differentiable transformation $g_\varphi(x, \varepsilon)$ of an auxiliary noise variable ε. In fact, g_φ is a mapping function which transforms a datapoint x and a random noise vector ε to a sample z. As a result, the modified variational lower bound is written as:

$$\begin{aligned} L^{VB}(\theta, \varphi, x) \;&=\; -D_{KL}(\, q_\varphi(z|x) \,\|\, p_\theta(z)\,) \;+\; E_{\,q_\varphi(z|x)}[\, log\, p_\theta(x|z)\,] \\ &=\; -D_{KL}(\, q_\varphi(z|x) \,\|\, p_\theta(z)\,) \;+\; \tfrac{1}{L} \sum_{l=1}^{L} [\, log\, p_\theta(x|z^l)\,] \end{aligned}$$

- $z^l = g_\varphi(x, \varepsilon^l)$ where $z^l \sim q_\varphi(z|x)$ and $\varepsilon^l \sim p(\varepsilon)$
- L: Number of sampling z^l for each datapoint. L = 1 if the size of minibatch is large enough.
- $D_{KL}(\, q_\varphi(z|x) \,\|\, p_\theta(z)\,)$: KL-divergence acts as a regularizer.

- $\frac{1}{L}\sum_{l=1}^{L}[\ log\, p_\theta(x|z^l)$: Expected negative reconstruction error, probability density of a datapoint x.

   i. Three approaches for a chosen approximate posterior $q_\varphi(z|x)$: A differentiable transformation $g_\varphi(x,\varepsilon)$ and a distribution for random noise vector $\varepsilon^l \sim p(\varepsilon)$
      - Tractable inverse CDF:
         - $g_\varphi(x,\varepsilon)$ is the inverse CDF of $q_\varphi(z|x)$
         - $\varepsilon \sim \upsilon(0, I)$
         - Example: Exponential, Cauchy, Logistic, Rayleigh, Pareto, Weibull, Reciprocal, Gompertz, Gumbel and Erlang distributions.
      - Location-Scale:
         - $g_\varphi(x,\varepsilon) =\ location\ +\ scale\ * \varepsilon$
         - ε: Auxiliary variable
         - Example: Laplace, Elliptical, Student's t, Logistic, Uniform, Triangular and Gaussian distributions
         - Univariate Gaussian:
            - $z\ \sim q_\varphi(z|x) = N(\mu,\ \sigma^2)$
            - $z\ = \mu\ +\ \sigma * \varepsilon)$
            - $\varepsilon \sim N(0,1)$
      - Composition: Random variables as different transformations of auxiliary variables.
         - Examples: Log-Normal (exponentiation of normally distributed variable), Gamma (a sum over exponentially distributed variables), Dirichlet (weighted sum of Gamma variates), Beta, Chi-Squared, and F distributions.

b. Language modeling

[Miao et al] proposes an unsupervised generative model of text. In this case, the MLP encoder compresses the bag-of-words document representation into a continuous latent distribution and the Softmax decoder reconstructs the document by generating word from the dense continuous latent distribution. It is based on the Location-Scale approach for approximate posterior (Gaussian distribution).

i. Neural Variational Inference Framework in [Miao et al]

[Kingma et al] uses the prior distribution over the hidden variable $p_\theta(z)$ as the standard Gaussian prior while [Miao et al] uses a conditional parameterised Gaussian distribution $p_\theta(z|x,y)$.

$$L = E_{q_\varphi(z|x)}[\ log\, p_\theta(x|z)\ ] - D_{KL}[q_\varphi(z|x) \,||\, p_\theta(z)\ ]$$ [Kingma et al]

$$L = E_{q_\varphi(z|x,y)}[log\, p_\theta(y|z)\, p_\theta(z|x)\, p(x) - log\, q_\varphi(z|x,y)]$$ [Miao et al]

- x: Input
- y: Output
- z: Hidden code

- $q_\varphi(z|x,y)$ in Gaussian distribution $N(\ z\ |\ [\mu(x,y),\ diag(\sigma^2(x,y))]\ )$
    - Construct vector representation of input and output:
      $$u = f_x(x)\ and\ v = f_y(y)$$
        - $f\,()$: Deep neural network for transformation
    - Assemble a joint representation
      $$\pi = g(u,v)$$
        - $g()$: MLP concatenates the vector representations
    - Parameterise the variational distribution over the latent variable z^l:
      $$z^l = \mu + \sigma * \varepsilon^l$$
      $$\mu = l_1(\pi)\ and\ \sigma = l_2(\pi)$$
        - $l_1()$ and $l_2()$: Linear transformation returns the parameter of Gaussian distribution for each input.
        - $\varepsilon^l \sim N(0,I)$
    - Gradient for back-propagation with sampling ε L times:
        - $\nabla_\mu L = \frac{1}{L}\sum_{l=1}^{L} \nabla_{z^l}[s(z^l)]$
        - $\nabla_\sigma L = \frac{1}{2L}\sum_{l=1}^{L} \varepsilon^l * \nabla_{z^l}[s(z^l)]$
        - $s(z^l) = log\, p_\theta(y|z^l)\, p_\theta(z^l|x) - log\, q_\varphi(z^l|x,y)$

ii. Language model of [Miao et al]

$$L = E_{q_\varphi(z|X)}[\sum_{i=1}^{N} log\, p_\theta(x_i|z)\ ] - D_{KL}[\ q_\varphi(z|X) \,||\, p_\theta(z)\ ]$$

- X: a document and
- x_i: a word in X
- N: Number of words in the document

- $p_\theta(x_i|z) = \frac{exp(-E(x_i;z,\theta)\ )}{\sum_{j=1}^{|Vocabulary|} exp(-E(x_j;z,\theta)\ )}$: Softmax decoder generating $z \rightarrow \{x_i\}$
    - $exp(-E(x_i;z,\theta)\ ) = -h^T R\, x_i - b_{x_i}$
    - R: Semantic word embeddings
    - b_{x_i}: Bias term
- $q_\varphi(z|X) = N(\ z|\ \mu(X),\ diag(\sigma^2(X))\ )$: MLP encoder without output.

   - $\mu = l_1(\pi)\ and\ \sigma = l_2(\pi)$
   - $\pi = g(u)$
   - $u = f_x(X)$
- $p_\theta(z)$: Gaussian prior for z

iii. Language model of [Bowman et al]

[Bowman et al] uses the same language model of [Miao et al]. However they proposed some versions for this architecture. They recognize that there is little difference between these versions. They proposed also the use of highway network layers with Deep Feedforward Network.

- Concatenating the sampled z to the decoder input at every time step
- Using a softplus parametrization for the variance
- Using Deep Feedforward Networks between the encoder and latent variable and the decoder and latent variable

[Bowman et al] has a problem of **Zero KL-divergence term** ($q_\varphi(z|x) = p_\theta(z)$), which means that the model does not encode the information of x. They explained that the problem comes from the sensitivity of the decoder. In other words, the (probably better) decoder ignores the encoder (Global latent variables) and try to predict itself the correct output sentences. Two examples mentioned are in image processing: [Kingma et al] uses a much weaker independent pixel decoder and a LSTM decoder [Theis et al] has a good performance without VAE global latent variables.

Two approaches for Zero KL-divergence:

- KL cost annealing: A variable weight δ is added for KL-divergence and is increased gradually from 0 to 1. At 1, the weight cost function is equivalent to the true variational lower bound.
$$L = E_{q_\varphi(z|x)}[\ log\, p_\theta(x|z)\ ] - \delta * D_{KL}[q_\varphi(z|x) \,||\, p_\theta(z)\ ]$$

- Word dropout and decoding without history: This approach is for weakening the decoder by removing some or all of conditioning information. In detail, the model replaces some fraction of the conditioned-on word tokens with the generic unknown word token (Do not use the true previous token). This technique is parameterised by a keep rate [0,1]. It is similar to Scheduled Sampling [Bengio et al].

Inputless decoder which does not condition on previous tokens (Word dropout keep rate of 0), is weaker than a standard decoder but its sentence generating process is fully differentiable. The differentiability is a property necessaire for adversarial training method.

c. Approach for NMT [Zhang et al]

There are three components including Encoder, Decoder ($p_\theta(y|z,x)$) and Inferer ($p_\theta(z|x)$ and $q_\varphi(z|x,y)$). In this case, the role of Encoder is encoding for both source and target sentence. The Decoder receives the hidden code z from Inferer. One point important is that the Inferer could model both the prior $p_\theta(z|x)$ and the posterior $q_\varphi(z|x,y)$.

The variational lower bound becomes:

$$L = E_{q_\varphi(z|x,y)}[\ log\, p_\theta(y|z,x)\ ] - D_{KL}[q_\varphi(z|x,y) \,||\, p_\theta(z,x)\ ]$$ [Zhang et al]

The difference between the equations of [Kingma et al] and [Miao et al] - [Zhang et al] is the KL-divergence. [Zhang et al] and [Miao et al] use VAE in their applications having the output i.e Translated sentence [Zhang et al] and Answer (Question-Answer System) [Miao et al]. [Kingma et al] with the application of language modeling, does not have the different output (The output is the input).

$$L = E_{q_\varphi(z|x)}[\ log\, p_\theta(x|z)\ ] - D_{KL}[q_\varphi(z|x) \,||\, p_\theta(z)\ ]$$ [Kingma et al]

i. Variational Neural Encoder

[Zhang et al] applies Bidirectional RNN. It notes that this encoder transforms both the source sentence and the target sentence. These representation vectors have two different dimensions for source s and target t. The source annotation flows into the Neural Prior Model and the Decoder. The target annotation flows into the Neural Posterior Approximator.

- From The source sentence $\{x_i\}_{i=1}^{T_s}$ to The source annotation vector $\{h_i\}_{i=1}^{T_s} \in R^{2d_s}$
- From The target sentence $\{y_i\}_{i=1}^{T_t}$ to The target annotation vector $\{h'_i\}_{i=1}^{T_t} \in R^{2d_t}$

ii. Variational Neural Inferer

The architecture of [Zhang et al]:

- Neural Posterior Approximator

$$q_\varphi(z|x,y) = N(z;\ \mu(x,y),\ \sigma(x,y)^2 I)$$

  - Step 1 (Mean-Pooling): $h_s = \frac{1}{T_s}\sum_i^{T_s} h_i$ and $h_t = \frac{1}{T_t}\sum_i^{T_t} h'_i$
  - **Step 2:** $h'_z = g(W_z^{(1)}[h_s : h_t] + b_z^{(1)})$ with g() is an element-wise activation function (tanh())
  - **Step 3:** $\mu = W_\mu h'_z + b_\mu$ and $log\sigma^2 = W_\sigma h'_z + b_\sigma$
  - Step 4: $h_z = \mu + \sigma \odot \varepsilon$, $\varepsilon \sim N(0, I)$

- Step 5: $h'_t = g(W_z^{(2)} h_z + b_z^{(2)})$ for Decoder

- Neural Prior Model

$$p_\theta(z|x) = N(z;\ \mu'(x),\ \sigma'(x)^2 I)$$

  - Step 1 (Mean-Pooling): $h_s = \frac{1}{T_s} \sum_i^{T_s} h_i$
  - **Step 2:** $h'_z = g(W_z^{(1)}[h_s] + b_z^{(1)})$
  - **Step 3:** $\mu' = W_{\mu'} h'_z + b_{\mu'}$
  - Step 4: $h_z = \mu'$ (Mean of $p_\theta(z|x)$)
  - Step 5: $h'_t = g(W_z^{(2)} h_z + b_z^{(2)})$ for Decoder

iii. Variational Neural Decoder

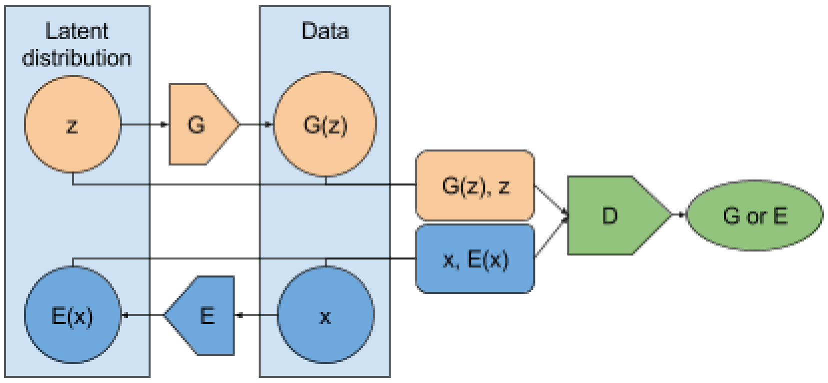
$$p_\theta(y|z,x) = \prod_{j=1}^{T_t} p(y_j|y_{<j}, z, x) = \prod_{j=1}^{T_t} g'(y_{j-1},\ s_{j-1}, c_j)$$

- g': Feed forward model
- Context $c_j = \sum_i \alpha_{ij} h_i$
- Hidden state s_j includes h'_t from the Prior Model Inferrer in the GRU unit.
  - $s_j = (1 - u_j) \odot s_{j-1} + u_j \odot \bar{s}_j$
  - $\bar{s}_j = tanh(W E_{y_j} + U[r_j \odot s_{j-1}] + C c_j + V h'_t)$ where E_{y_j} is word embedding of target sentence
  - $u_j = \sigma(W_u E_{y_j} + U_u s_{j-1} + C_u c_j + V_u h'_t)$
  - $r_j = \sigma(W_r E_{y_j} + U_r s_{j-1} + C_r c_j + V_r h'_t)$

D. GANs and Continuous Space in Global Feature Learning [Donahue et al]

[Donahue et al] underlines the role of semantic variation in the GAN framework. In GANs, the Generator generates samples from a latent distribution and learns to map this simple distribution to a more realistic and complex data distribution. The Discriminator evaluates these samples and not directly the latent distribution. In the case that the data semantics are relevant, projecting data back into the latent distribution or learning the feature representation could be an approach remarkable. As a result, the role of Discriminator is not only distinguish these samples (Real or Fake) but their latent distributions. They hypothesize that the back-transformation may serve as a useful feature representation for related semantic tasks. Therefore, [Donahue et al] proposes Bidirectional Generative Adversarial Networks (BiGAN) for the application of unsupervised feature representation (Classification in processing image).

In detail, there are three components: Generator G, Discriminator D and Encoder E. This framework based on GAN, has a new component Encoder E which maps data x to latent representation z. The Discriminator discriminates (x, E(x)) and (G(z), z) where G(z) and x are data, E(x) and z are in feature space (Latent distribution). The schema is below:



The two components G and E learns to invert one another to fool D. With regards to the schema above, G and E communicate indirectly ($x \sim p_x$ and $z \sim p_z$). The paper proposes also Latent Regressor, an alternative approach for baseline comparison where G and E communicate directly (E(G(z)) and G(E(x)).

The latent representation z may be thought of as a "label" for x without the need of supervisor. Therefore, [Donahue et al] concludes that BIGANs are a robust and highly generic approach to unsupervised feature learning, without the knowledge about the structure or the data type.

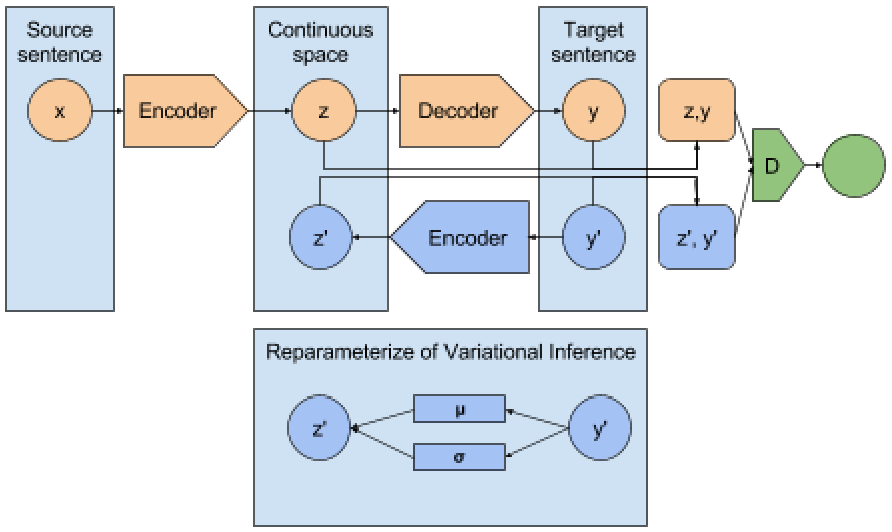
The objective function of BiGAN: $min_{G,E}\ max_D V(D,E,G)$

$$V(D,E,G) = E_{x\sim p_x}[\ E_{z\sim p_E(\cdot|x)}[log\ D(x,z)]\ ] + E_{z\sim p_z}[\ E_{x\sim p_G(\cdot|z)}[\ log\ (1 - D(x,z)\ )\ ]\ ]$$
$$= E_{x\sim p_x}[log\ D(x,E(x)\ )\ ] + E_{z\sim p_z}[\ log\ (1 - D(G(z),z)\ )\ ]$$

For Latent Regressor, the Encoder is trained only on the generated samples G(z) and not on the real data $x \sim p_x$. Its main drawback is that the Encoder does not represent the semantic feature of real data. The objective function is rewritten below:

$$V(D,E,G) = E_{z\sim p_z}[log\ D(G(z),E(G(z))\ )\ ] + E_{z\sim p_z}[\ log\ (1 - D(G(z),z)\ )\ ]$$

In the case of NMT, projecting the target sentence back into the latent continuous space could improve the translation ? An example schema is below**.**



- From the schema, It could be seen that D force z = z'. The system MT and Variational Encoder try to fool D.
    - How to pre-train D ?
    - We set the true target sentence or machine-generated sentence as the input of Variational Encoder ? If the input is true target sentence, we have the case of Human-Machine Discriminator.
- In the case we use Variational Inference, there are some points should be considered:
    - How to select the prior distribution (ex: Gaussian) ?
    - Are the continuous space of z and z' are similar ?
- **We could recognize that the schema is similar to Dual-Learning. However, this approach concentrates not only on the discrete sequence but on the continuous space.**

E. GAN and Variational Training in Energy Based Models [Zhai et al]

[Zhai et al] proposes Variational GAN (VGAN). For the Energy Based Models (EBMs), VGAN minimizes the variational lower bound of its negative log likelihood. The purpose is predicting the energy of a sample.

- The model density p(x) is approximated by a variational distribution q(x).
- For GANs, p(x) corresponds to Discriminator, q(x) corresponds to Generator.
- Training of VGANs has two steps:
    - Given p(x), q(x) is updated to maximize the lower bound.
    - p(x) is updated one step with samples drawn from q(x) to decrease the lower bound.

[Zhai et al] mentioned the ability significant of GAN and EBM in unsupervised learning because of its minimal assumptions (EBM) or non-assumption (GAN) about data distribution. The assumption of EBM is the energy of a sample (The negative log density E(x) = - log p(x) as a deterministic function of x). Therefore, [Zhai et al] combines two kinds of models where

- The energy E(x) = -log D(x). D(x) is from Discriminator. It is a viable solution for the MLE of EBMs. (From GAN view)
- $p(x) = \frac{e^{-E(x)}}{\int_x e^{-E(x)}dx}$ in the generator G(z) or a parameterized sampler. (From EBM view)
- $p_z(\bar{x}\,|x)$ instead of $p_z(x)$. It parameterize the generator as a transition distribution (From Variational view).

D is trained to increase D(x) for $x \sim p_{data}(x)$ and decrease D(x) for $x \sim p_g(x)$. G is trained to increase D(x) for $x \sim p_g(x)$. Therefore, the objective function of VGAN:

$$min_E \; max_G \; E_{x\sim p_{data}(x)}[\, E(x)\,] - E_{x\sim p_g(x)}[\, E(x)\,] + H(p_g)$$
$$= min_D \; max_G \; E_{x\sim p_{data}(x)}[- \, logD(x)\,] - E_{z\sim p_z(z)}[\, - \, log\, D(G(z))\,] + H(p_g)$$

- $H(p_g)$: Entropy term of p_g (Arbitrary distribution)
- G(z): Sample generated by Generator from implicit data distribution ($z \sim p_z(z)$)
- D(x): Probability of x being from Generator or True data

Compared with GANs, three aspects in VGAN:

- The order of minimization and maximization:
    - The outcome of GAN is Generator
    - The outcome of VGAN is Discriminator (Predicting the energy) and G can sample from the exact distribution defined by D.
- The parameterization of energy:
    - The energy parameterization of GANs is lower bounded by 0.
    - The energy of EBM has no lower bound, which could cause the negative infinity of energy.
- The entropy term of the generator distribution: It could help Generator to not fall into one of the few local minima of the energy landscape. As a result, an other challenge is the approximation of this term.

F. Reference

a. [Bowman et al]: Generating Sentences From A Continuous Space
b. [Donahue et al]: Adversarial Feature Learning
c. [Miao et al]: Variational Inference for Text Processing
d. [Kingma et al]: Auto-Encoding Variational Bayes
e. [Zhang et al]: Variational Neural Machine Translation
f. [Zhai et al]: Generative Adversarial Networks As Variational Training Of Energy Based Models