[Paper review 35]

A Stochastic Decoder for Neural Machine Translation

(Schulz, et al., 2018)

[Contents]

1. Abstract

present a deep generative model of **machine translation**, which incorporates **"a chain of LATENT VARIABLE"**

2. Introduction

Machine Translation

- based on encoder-decoder framework, complex neural systems are being developed!
- ex) use of convolutions, self-attention layers...
- great performance improvements over classical RNNs!

But there hasn't been much effort to change the **probabilistic model**

- ex) sentence-level latent Gaussian variable (Zhang et al, 2016)

Not only does translation may vary across translators, but also within a single translator!

But NMT are incapable of capturing these variations!

- only one output for a given source sentence
- $P\left(y_1^n \mid x_1^m\right)=\prod_{i=1}^n P\left(y_i \mid x_1^m, y_{<i}\right)$.

Proposal of this paper : augment NMT with **"latent sources of variation"**
(to be able to represent more of the variation)

Contribution

- introduce NMT that is capable of capturing **word level variation**
- motivate the use of **KL scaling**
- improvements achievable with the proposed model

3. Neural Machine Translation

likelihood : $P\left(y_1^n \mid x_1^m\right)=\prod_{i=1}^n P\left(y_i \mid x_1^m, y_{<i}\right)$.

notation

- source sentence : $x_1^m=\left(x_1, \ldots, x_m\right)$.
- target sentence : y_1^n.
- Encoder : bi-LSTM
  Decoder : LSTM
- decoder state at the i^{th} target position : t_i

How does it work?

$$
\begin{aligned}
{\left[h_1, \ldots, h_m\right] } & =\operatorname{RNN}\left(x_1^m\right) \\
\tilde{t}_i & =\operatorname{RNN}\left(t_{i-1}, y_{i-1}\right) \\
e_{ij} & =v_a^{\top} \tanh \left(W_a\left[\tilde{t}_i, h_j\right]^{\top}+b_a\right) \\
\alpha_{ij} & =\frac{\exp \left(e_{ij}\right)}{\sum_{j=1}^m \exp \left(e_{ij}\right)} \\
c_i & =\sum_{j=1}^m \alpha_{ij} h_j \\
t_i & =W_t\left[\tilde{t}_i, c_i\right]^{\top}+b_t \\
\phi_i & =\operatorname{softmax}\left(W_o t_i+b_o\right)
\end{aligned}
$$

- trained using **MLE**
  - loss function : **cross entropy**
  - probability vector : by **softmax**

4. Stochastic Decoder

introduce stochastic decoder model for capturing **word-level variation**

4-1. Motivation

Even within a single translator, **variation may occur!**

Previous work : modeling the latent variation (using sentence-level Gaussian Variable)

- however there is more to latent variation than a unimodal density can capture
- "Multimodal modelling" of these variation is needed!

$\rightarrow$ consider **word level** variation

4-2. Model formulation

"latent Gaussian variable" for each target position

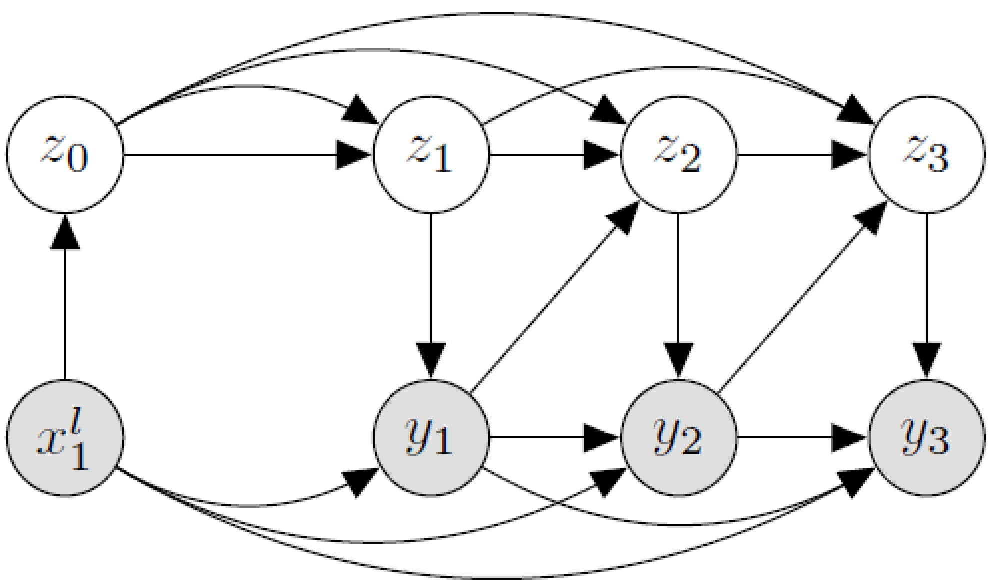
- depends on...
  - (1) previous latent states
  - (2) decoder state

Thus, the likelihood could be written as....

$P\left(y_1^n \mid x_1^m\right)=\int p\left(z_0 \mid x_1^m\right) \mathrm{d} z_0^n \times \quad \prod_{i=1}^n p\left(z_i \mid z_{<i}, y_{<i}, x_1^m\right) P\left(y_i \mid z_1^i, y_{<i}, x_1^m\right)$

- contains z_0 (= 0^{th} latent variable) , which is meant to initialize the chain of latent variables based solely on the source sentence

  (previous sentence based model ONLY used that term!)



(a)

- stochastic decoder model

  (= generator model)

$$\begin{aligned} Z_0 \mid x_1^m & \sim \mathcal{N}\left(\mu_0, \sigma_0^2\right) \\ Z_i \mid z_{<i}, y_{<i}, x_1^m & \sim \mathcal{N}\left(\mu_i, \sigma_i^2\right) . \\ Y_i \mid z_0^i, y_{<i}, x_1^m & \sim \operatorname{Cat}\left(\phi_i\right) \end{aligned}$$

- μ and σ are predicted by NN architecture

4-3. Neural Architecture

It is DGM (deep generative models)

- $\because$ model contains latent variable & parameterized by NN
- use reparameterization trick!
  - to enable back-prop inside a stochastic computation graph
  - $z=\mu+\sigma \odot \epsilon \quad \epsilon \sim \mathcal{N}(0, \mathrm{I})$.

Structure

- one-hidden layer NN
- activation function : tanh
- softplus transformation to the output of the standard deviation's network (for positivity)
- $\mu_{0}=f_{\mu_{0}}\left(h_{m}\right) \quad \sigma_{0}=f_{\sigma_{0}}\left(h_{m}\right)$.

  $\mu_{i}=f_{\mu}\left(t_{i-1}, z_{i-1}\right) \quad \sigma_{i}=f_{\sigma}\left(t_{i-1}, z_{i-1}\right) \quad .$

Each latent variable is sampled by $z=\mu+\sigma \odot \epsilon \quad \epsilon \sim \mathcal{N}(0, \mathrm{I})$

- then, used to modify the update of decoder hidden state t_i

  $\tilde{t}_{i}=\operatorname{RNN}\left(t_{i-1}, y_{i-1}, z_{i}\right)$.

5. Inference and Training

use Variational Inference to train the model

(= maximization of ELBO)

ELBO is maximized w.r.t

- model parameters θ (= parameter of $p(x)$)
- variational parameters λ (= parameter of $q(z)$)

NLP models using DGMs

- mostly use only ONE latent variable
- using several variables : MFVI (assumption : independency between latent variables)
- this paper : more FLEXIBLE (assign dependency)

  $q(z_0^n)=\prod_{i=1}^{n} q\left(z_{i} \mid z_{<i}\right)$.

Stochastic decoder of this paper = **"Stack of conditional DGMs"**

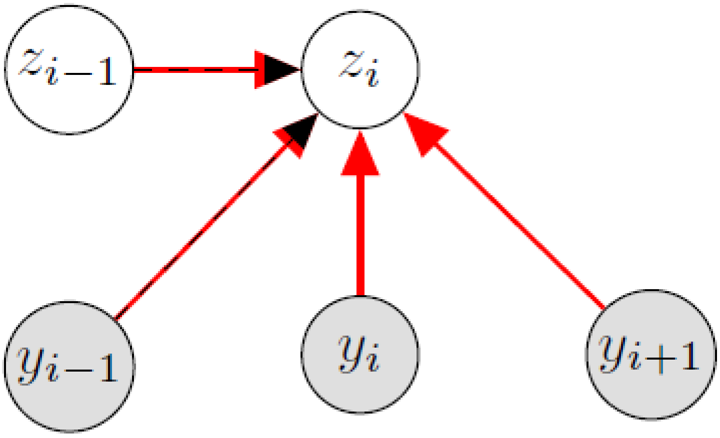
(thus consists of nested positional ELBOS)

$\mathrm{ELBO}_0+\mathbb{E}_{q(z_0)}\left[\mathrm{ELBO}_1+\mathbb{E}_{q(z_1)}\left[\mathrm{ELBO}_2+\ldots\right]\right]$.

- where target position i in ELBO is
  $\mathrm{ELBO}_i=\mathbb{E}_{q(z_i)}\left[\log p\left(y_i \mid x_1^m, y_{<i}, z_{<i}, z_i\right)\right]-\mathrm{KL}\left(q\left(z_i\right) \| p\left(z_i \mid x_1^m, y_{<i}, z_{<i}\right)\right)$.
  - first term : **reconstruction** or **likelihood** term
  - second term : KL term (= function of 2 Gaussians .... can be solved analytically)

Inference model

- use NN to compute variational distributions
- (during training) both source & target are observed
- z_i .
  - 1) condition on information available to the generation network



  - 2) condition on the target words

    ($\tilde{t}_i=\operatorname{RNN}(t_{i-1}, y_{i-1}, z_i)$)
- produces additional representations of the target sentence
  - 1st rep) encodes the target sentence bidirectionally
  - 2nd rep) encoding the target sentence in reverse

  $$\begin{aligned}&[b_1, \ldots, b_n]=\operatorname{RNN}(y_1^n) \\ &[r_1, \ldots, r_n]=\operatorname{RNN}(y_1^n)\end{aligned} .$$
- same as generative model....
  - also use one-hidden layer NN
  - each latent variable is sampled by $z=\mu+\sigma \odot \epsilon \quad \epsilon \sim \mathcal{N}(0, \mathrm{I})$

    $$\begin{aligned}\mu_0 &=g_{\mu_0}\left(h_m, b_n\right) \\ \sigma_0 &=g_{\sigma_0}\left(h_m, b_n\right) \\ \mu_i &=g_\mu\left(t_{i-1}, z_{i-1}, r_i, y_i\right) \\ \sigma_i &=g_\sigma\left(t_{i-1}, z_{i-1}, r_i, y_i\right)\end{aligned} .$$

  (during training, all samples are sampled from inference network)

  (sample from the generator only at the test time!)

5-1. Analysis of the Training Procedure

does not work well in practice... WHY?

∵ our model use a STRONG generator

(= do not need to rely on latent information)

(Can be understood by the KL-term below)

For latent-variable to be informative, we should have high mutual information :

$I(Z;Y) = \mathbb{E}_{p(y)}[\mathrm{KL}(p(Z \mid Y) \| p(Z))]$.

$I(Z;Y) = \mathbb{E}_{p(y)}[\mathrm{KL}(p(Z \mid Y) \| p(Z))]$.

- we approximate $p(Z \mid Y)$ with $q(Z \mid Y)$
- KL-term in ELBO is upper bound on mutual information
- ELBO can be maximized, by (1) setting KL-term to 0 and (2) maximizing reconstruction term

  $\rightarrow$ ∴ (at the beginning of training) variational approximation does not yet encode much useful information!

  (= during the initial learning stage, KL-term barely contributes to ELBO (our objective))