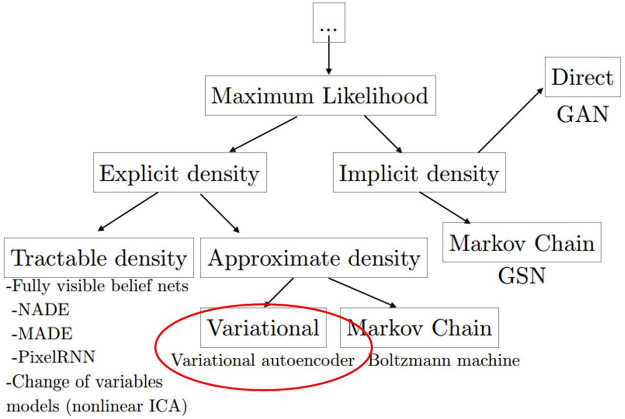
Lagging inference networks and posterior collapse in variational autoencoders

2020/09/21 (Mon)
논문 리뷰
김민지

Taxonomy of deep generative models



ML 가능도를 최대화하는 방법에서의 생성 모델들

Explicit density 확률 모델을 정의하고, 이를 최대화

1. 모델을 정의했기 때문에 다루기가 편하고
2. 모델의 움직임을 예측하기 쉽지만
3. 우리가 아는 것 이상으로 결과를 끌어낼 수 없다.

Approximate density 정의한 확률 모델이 계산 불가능할 때, 이를 근사

ex. variational inference - VAE

https://arxiv.org/abs/1701.00160
http://jaejunyoo.blogspot.com/2017/04/auto-encoding-variational-bayes-vae-1.html

Variational AutoEncoder

Goal 데이터의 확률모델 P(x)을 학습하여 데이터를 **생성**
Method Variational Inference + AutoEncoder의 구조 (Encoder-Decoder) 를 사용하겠다

알고자 하는 것: P(x)
모든 z에 대한 결합 분포 P(x,z)의 적분으로 주변함수 P(x)

$$p(\mathbf{x}) = \int_z p(\mathbf{x}, \mathbf{z}) dz = \int_z p(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) dz$$

P(x)의 가능도를 최대화하는 모델 파라미터 θ 학습
하지만 모든 z를 알 수도 없고, p(z)도 모르기 때문에 **intractable**

$$p(x) = \int_z p(\mathbf{x}|\mathbf{z}; \theta)\, p(\mathbf{z})\, dz = \int_z p_\theta(\mathbf{x}|\mathbf{z})\, p(\mathbf{z})\, dz$$

Variational AutoEncoder

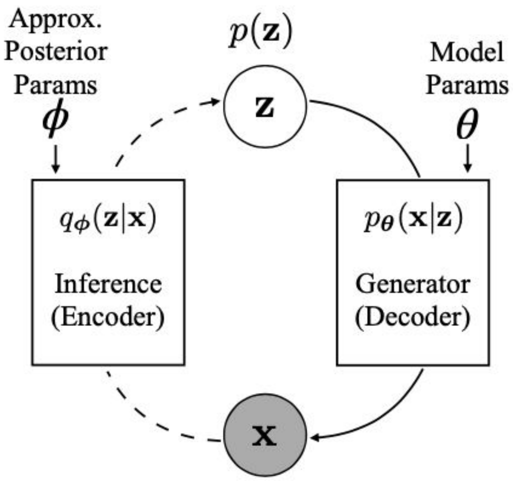
Intractable: $\boxed{p_\theta(\mathbf{z}|\mathbf{x})} = \dfrac{\boxed{p_\theta(\mathbf{x}|\mathbf{z})}\,p_\theta(\mathbf{z})}{p_\theta(\mathbf{x})}$

Encoder
잠재변수 z로부터 데이터 x
생성

Decoder
데이터 x의 feature를 z로 encoding

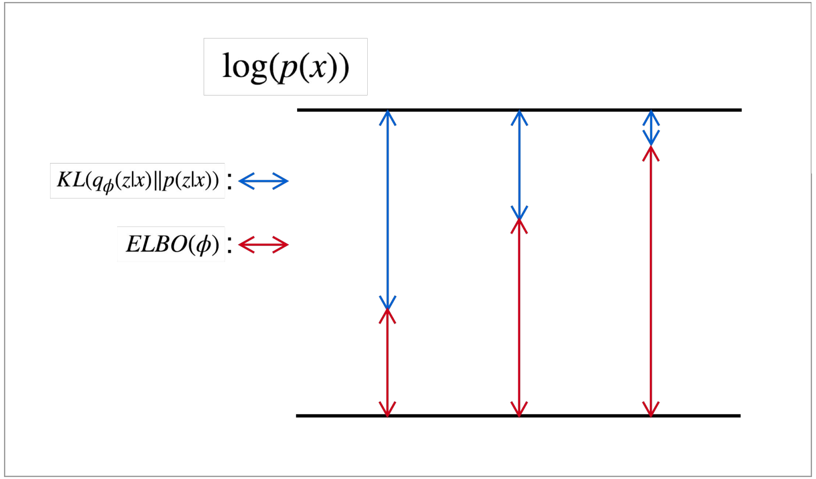
$$p(x) = \int_z p(\mathbf{x}|\mathbf{z};\theta)\ p(\mathbf{z})\ dz = \int_z p_\theta(\mathbf{x}|\mathbf{z})\ p(\mathbf{z})\ dz$$

Variational Approximation



(a) Variational autoencoders

Evidence Lower BOund (ELBO)



$$\begin{aligned}\mathcal{L}(\theta, \phi, \mathbf{x}) &= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x, z) - \log q_\phi(z|x)] \\ &= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z) + \log p_\theta(z) - \log q_\phi(z|x)] \\ &= -D_{\mathrm{KL}}[q_\phi(z|x)||p_\theta(z)] + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]\end{aligned}$$

D_{KL}이 계산 불가능한 경우가 많기 때문에,
lower bound를 증가시켜서 D_{KL}을 최소화할 수 있다.
=> 1. p(z)와 q(z|x)의 분포 차이를 최소화하면서
2. 모델(Decoder)로 z로부터 생성한 p(x|z)가
Eecoder q(z|x)가 encoding한 분포에서
기인했을 가능도를 maximize

Posterior collapse

: VAE의 가장 큰 단점은 decoder의 학습, (Inference network)이 힘들다

VAE의 decoder 학습 시그널을 생각해보면 reconstruction loss에만 의존한다.

문제는 학습 입력이 주어졌을 때, encoder의 결과가 z가 p(z) (대부분 가우시안) 분포를 따르기 때문에,

z이 값이 직접 decoder로 들어가는 것이 아닌, p(z)에서 sample된 값들이 decoder로 들어가게 된다.

그래서 decoder에서 나온 reconstructed input에 대한 recon loss가 쉽게 줄어들지 않게 된다.

Posterior collapse

$$\mathcal{L}(\theta,\phi,x) = \underbrace{\mathbb{E}_{z\sim q_\phi(z|x)}\log p_\theta(x|z)}_{\text{(i)}} - \underbrace{\mathrm{KL}[q_\phi(z|x)||p(z)]}_{\text{(ii)}} \leq \log p_\theta(x) \quad (*)$$

Claim: Suppose that (i) there exists θ^* such that $p_{\theta^*}(x|z) = p_{\text{data}}(x)$ for all x, and (ii) there exists ϕ^* such that $q_{\phi^*}(z|x) = p(z)$ for all z. Then (θ^*, ϕ^*) is a globally optimal solution to the VAE objective.

Proof: If $p_{\theta^*}(x|z) = p_{\text{data}}(x)$ then $p_{\theta^*}(z|x) = p(z)$, and thus $\mathrm{KL}[p_{\theta^*}(z|x)||q_{\phi^*}(z|x)] = 0$ and so the variational lower bound in Equation $(**)$ is tight. That is,

$$\begin{aligned}\log p_{\theta^*}(x) &= \mathcal{L}(\theta^*,\phi^*,x)\\ &= \mathbb{E}_{z\sim q_{\phi^*}(z|x)}[\log p_{\theta^*}(x|z)] + \mathrm{KL}[q_{\phi^*}(z|x)||p(z)]\\ &= \log p_{\text{data}}(x)\end{aligned}$$

Thus the objective of the VAE is at its global optimum. □

http://paulrubenstein.co.uk/variational-autoencoders-are-not-autoencoders/

Posterior collapse

x -> z mapping을 사용하지 않고 바로 prior인 p(z)를 따르게 될 때.

$$q_\phi(\mathbf{z}|\mathbf{x}) = p_\theta(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}) \text{ for all } \mathbf{x}.$$

$$\boxed{p_\theta(\mathbf{z}|\mathbf{x}) = p(\mathbf{z})}$$ Model Collapse

$$\boxed{q_\phi(\mathbf{z}|\mathbf{x}) = p(\mathbf{z})}$$ Inference Collapse

Intuitions from ELBO

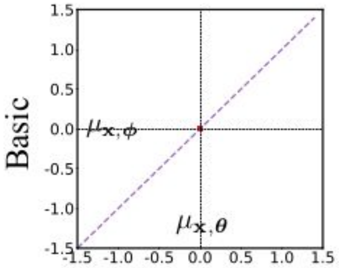
$$\mathcal{L}(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \underbrace{\log p_{\boldsymbol{\theta}}(\mathbf{x})}_{\text{marginal log data likelihood}} - \underbrace{D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \| p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))}_{\text{agreement between approximate and model posteriors}} ;$$

$$\nabla_{\boldsymbol{\theta}} D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) | p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})) = 0 \text{ when } q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) = p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}).$$

q(z|x)관점에서는 p(z|x)와 일치하는 것이 유일한 목표.
p(z|x)관점에서는 marginal data likelihood, q(z|x)로부터 얻어지는 p(z|x)

$$\mathcal{L}(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\phi}) = \underbrace{\log p_{\boldsymbol{\theta}}(\mathbf{x})}_{\text{marginal log data likelihood}} - \underbrace{D_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})\|p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))}_{\text{agreement between approximate and model posteriors}} ;$$

Intuitions from ELBO



1. 학습 초기에서 z와 x는 q(z|x)와 p(z|x) 모두에서 독립적이기 때문에, 모든 x에서는 model collapse를 겪게 된다 $\boxed{p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) = p(\mathbf{z})}$

2. 목적함수에서, p(z|x)에서 z,x 사이의 의존성을 야기하는 것이 marignal log data likelihood logp(x) 밖에 존재하지 않게된다.

3. 그런데 z, x가 독립적인 상태로 두 분포가 갈라지기 시작하면 D_{KL}에 의해 압도될 수 있다.

4. 우리는 실제로 학습이 p(z|x)와 q(z|x)를 prior p(z)로 유도하여 정렬하도록하는한편, z를 무시하면서 x의 분포를 캡처하는 모델 매개 변수에 고정한다고 가정한다.

5. 문제는 이러한 posterior collapse가 local optimum이고, z를 사용하여 x를 설명하는 더 좋은 모델이 있음에도 최적화에 실패하게 된다.

Observations on Synthetic Data

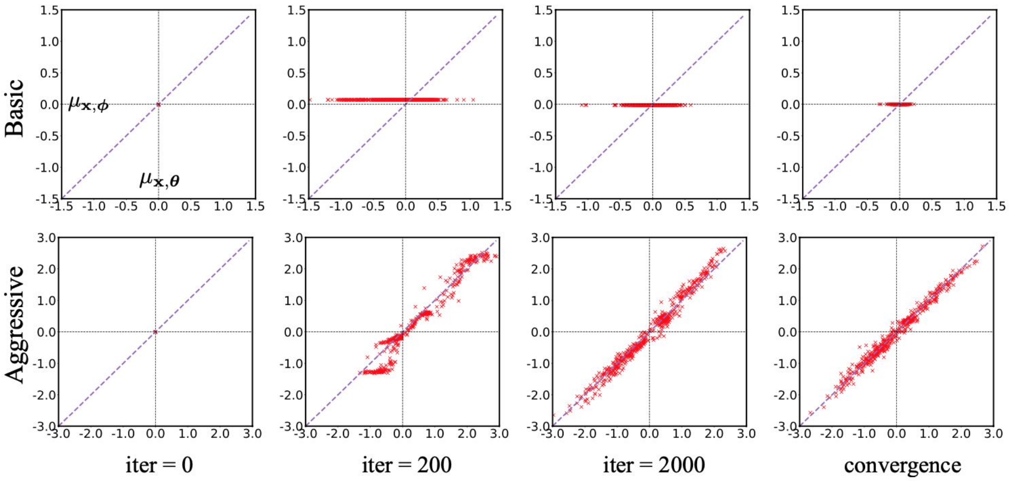


Figure 2: The projections of 500 data samples from a synthetic dataset on the posterior mean space over the course of training. "iter" denotes the number of updates of generators. The top row is from the basic VAE training, the bottom row is from our aggressive inference network training. The results show that while the approximate posterior is lagging far behind the true model posterior in basic VAE training, our aggressive training approach successfully moves the points onto the diagonal line and away from inference collapse.

1. 원점에서 시작
   q, p 모두에서 거의 독립적이다
2. μ x,θ 축을 따라 퍼진다
   p(z)로부터 멀리 떨어진 데이터 생성, logp(x)가 model collapse를 방지한다.
3. 수평선에만 존재한다
   q(z|x)가 p(z|x)를 따라잡는데 실패했다 (inference collapse)
4. 예상과 같이, p(z|x)에서의 z와 x의 의존성은 점점 줄어들고, collapsed local optimum에 수렴하게 된다

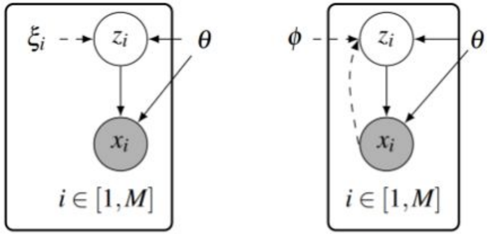
Aggressive training of the inference network

inference network q(z|x)가 p(z|x)보다 뒤쳐지기 때문에 발생.
inference network를 “aggressive"하게 업데이트하는것이 필요하다.
=> 두 개의 업데이트를 따로 떨어트려 최적화하는 방법 선택.

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi}^*), \text{ where } \boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}} \mathcal{L}(\mathbf{X}; \boldsymbol{\theta}, \boldsymbol{\phi}),$$

Stopping Criterion

이런식의 학습은 amortized inference network의 이점을 무시하는 것과 같다.



(a) SVI (b) VAE

D_{KL}은 q(z|x)나 p(z|x)중 하나가 p(z)와 가까울 때 하나만 p(z)로 밀어부치는 경향이 있다.
따라서 이러한 상태에 도달하지 않았음을 확인할 수 있다면, standard VAE training이 가능하다.

https://arxiv.org/pdf/1711.05597

Observations on Synthetic Dataset

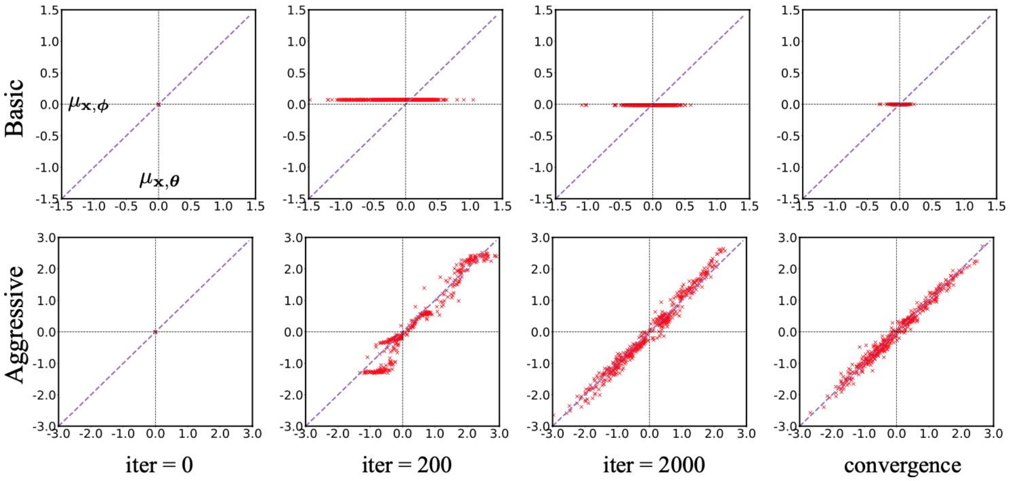


Figure 2: The projections of 500 data samples from a synthetic dataset on the posterior mean space over the course of training. “iter” denotes the number of updates of generators. The top row is from the basic VAE training, the bottom row is from our aggressive inference network training. The results show that while the approximate posterior is lagging far behind the true model posterior in basic VAE training, our aggressive training approach successfully moves the points onto the diagonal line and away from inference collapse.

데이터 포인트가 $\mu\ x,\theta = \mu\ x,\varphi$ 로 이동하고, 수렴상태에서는 대각선을 따라 이동한다.

이는 inference network을 더 update하는 쉬운 방법으로 inference -generator optimization이 균형을 이룰 수 있음을 보여준다.

Algorithm 1 VAE training with controlled aggressive inference network optimization.

```
θ, ϕ ← Initialize parameters
aggressive ← TRUE
repeat
    if aggressive then
        repeat                                          ▷ [aggressive updates]
            X ← Random data minibatch
            Compute gradients g_ϕ ← ∇_ϕ L(X; θ, ϕ)
            Update ϕ using gradients g_ϕ
        until convergence
        X ← Random data minibatch
        Compute gradients g_θ ← ∇_θ L(X; θ, ϕ)
        Update θ using gradients g_θ
    else                                                ▷ [basic VAE training]
        X ← Random data minibatch
        Compute gradients g_{θ,ϕ} ← ∇_{ϕ,θ} L(X; θ, ϕ)
        Update θ, ϕ using g_{θ,ϕ}
    end if
    Update aggressive as discussed in Section 4.2
until convergence
```

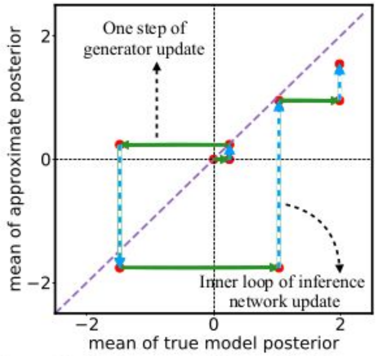


Figure 3: Trajectory of one data instance on the posterior mean space with our aggressive training procedure. Horizontal arrow denotes one step of generator update, and vertical arrow denotes the inner loop of inference network update. We note that the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ takes an aggressive step to catch up to the model posterior $p_\theta(\mathbf{z}|\mathbf{x})$.

inference collapse로부터 벗어나게 해준다.

Relation to Related Work

1. KL cost annealing (정규화 관점)
   처음에는 recon loss만 사용하고, KL 항의 가중치(베타β)가 "warm up"기간에 작은 값에서 점점 증가한다.
   - KL 가중치를 하이퍼 파라미터 (e.g. β)
   - KL 의 최소값을 제한하는 "free bits"
   - ...
2. Amortization Gap
   Inference 네트워크의 parameter 공유로 인한 ELBO의 차이
   - Instance-specific variational inference와 결합하여 이 gap을 줄인다.
   - e.g. SA-VAE

Table 1: Results on Yahoo and Yelp datasets. We report mean values across 5 different random restarts, and standard deviation is given in parentheses when available. For LSTM-LM* we report the exact negative log likelihood.

	Yahoo				Yelp			
Model	**NLL**	**KL**	**MI**	**AU**	**NLL**	**KL**	**MI**	**AU**
Previous Reports								
CNN-VAE (Yang et al., 2017)	$\leq$332.1	10.0	–	–	$\leq$359.1	7.6	–	–
SA-VAE + anneal (Kim et al., 2018)	$\leq$327.5	7.19	–	–	–	–	–	–
Modified VAE Objective								
VAE + anneal	328.6 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	357.9 (0.1)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
β-VAE ($\beta = 0.2$)	332.2 (0.6)	19.1 (1.5)	3.3 (0.1)	20.4 (6.8)	360.7 (0.7)	11.7 (2.4)	3.0 (0.5)	10.0 (5.9)
β-VAE ($\beta = 0.4$)	328.7 (0.1)	6.3 (1.7)	2.8 (0.6)	8.0 (5.2)	358.2 (0.3)	4.2 (0.4)	2.0 (0.3)	4.2 (3.8)
β-VAE ($\beta = 0.6$)	328.5 (0.1)	0.3 (0.2)	0.2 (0.1)	1.0 (0.7)	357.9 (0.1)	0.2 (0.2)	0.1 (0.1)	3.8 (2.9)
β-VAE ($\beta = 0.8$)	328.8 (0.1)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	358.1 (0.2)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
SA-VAE + anneal	327.2 (0.2)	5.2 (1.4)	2.7 (0.5)	9.8 (1.3)	355.9 (0.1)	2.8 (0.5)	1.7 (0.3)	8.4 (0.9)
Ours + anneal	**326.7 (0.1)**	5.7 (0.7)	2.9 (0.2)	15.0 (3.5)	**355.9 (0.1)**	3.8 (0.2)	2.4 (0.1)	11.3 (1.0)
Standard VAE Objective								
LSTM-LM*	**328.0 (0.3)**	–	–	–	358.1 (0.6)	–	–	–
VAE	329.0 (0.1)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	358.3 (0.2)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
SA-VAE	329.2 (0.2)	0.1 (0.0)	0.1 (0.0)	0.8 (0.4)	357.8 (0.2)	0.3 (0.1)	0.3 (0.0)	1.0 (0.0)
Ours	328.2 (0.2)	5.6 (0.2)	3.0 (0.0)	8.0 (0.0)	**356.9 (0.2)**	3.4 (0.3)	2.4 (0.1)	7.4 (1.3)

Table 2: Results on OMNIGLOT dataset. We report mean values across 5 different random restarts, and standard deviation is given in parentheses when available. For PixelCNN* we report the exact negative log likelihood.

Model	NLL	KL	MI	AU
Previous Reports				
VLAE (Chen et al., 2017)	89.83	–	–	–
VampPrior (Tomczak & Welling, 2018)	89.76	–	–	–
Modified VAE Objective				
VAE + anneal	89.21 (0.04)	1.97 (0.12)	1.79 (0.11)	5.3 (1.0)
β-VAE ($\beta = 0.2$)	105.96 (0.38)	69.62 (2.16)	3.89 (0.03)	32.0 (0.0)
β-VAE ($\beta = 0.4$)	96.09 (0.36)	44.93 (12.17)	3.91 (0.03)	32.0 (0.0)
β-VAE ($\beta = 0.6$)	92.14 (0.12)	25.43 (9.12)	3.93 (0.03)	32.0 (0.0)
β-VAE ($\beta = 0.8$)	89.15 (0.04)	9.98 (0.20)	3.84 (0.03)	13.0 (0.7)
SA-VAE + anneal	**89.07 (0.06)**	3.32 (0.08)	2.63 (0.04)	8.6 (0.5)
Ours + anneal	89.11 (0.04)	2.36 (0.15)	2.02 (0.12)	7.2 (1.3)
Standard VAE Objective				
PixelCNN*	89.73 (0.04)	–	–	–
VAE	89.41 (0.04)	1.51 (0.05)	1.43 (0.07)	3.0 (0.0)
SA-VAE	89.29 (0.02)	2.55 (0.05)	2.20 (0.03)	4.0 (0.0)
Ours	**89.05 (0.05)**	2.51 (0.14)	2.19 (0.08)	5.6 (0.5)

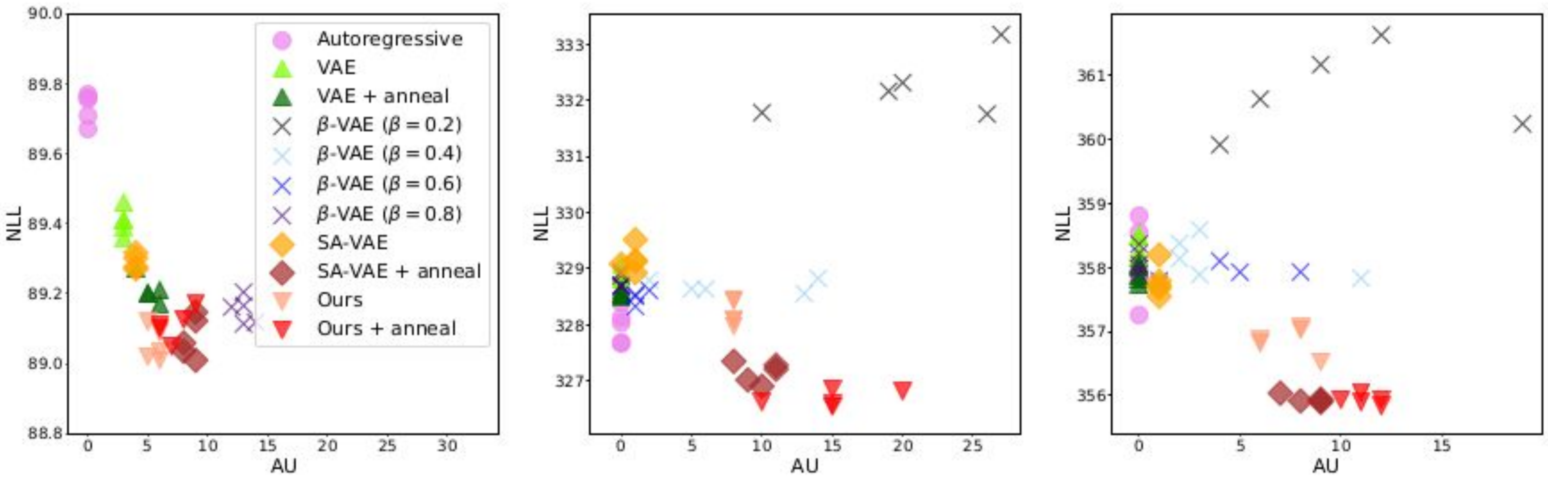
Experiments - Set up

Negative Log Likelihood - NLL (tighter lower bound)

DKL

Mutual Information

Active Units



(a) OMNIGLOT (b) Yahoo (c) Yelp

Figure 4: NLL versus AU (active units) for all models on three datasets. For each model we display 5 points which represent 5 runs with different random seeds. "Autoregressive" denotes LSTM-LM for text data and PixelCNN for image data. We plot "autoregressive" baselines as their AU is 0. To better visualize the system difference on OMNIGLOT dataset, for OMNIGLOT figure we ignore some β-VAE baselines that are not competitive.

Results

- PixelCNN에서 larger decoder 사용
- SA-VAE에서는 annealing 없이는 posterior collapse 발생

Table 3: Comparison of total training time, in terms of relative speed and absolute hours.

	Yahoo		Yelp15		OMNIGLOT	
	Relative	**Hours**	**Relative**	**Hours**	**Relative**	**Hours**
VAE	1.00	5.35	1.00	5.75	1.00	4.30
SA-VAE	9.91	52.99	10.33	59.37	15.15	65.07
Ours	2.20	11.76	3.73	21.44	2.19	9.42

Analysis of Baselines

- KL regularizer weakening과 비교
  1. mutual information between z and x, Iq
  2. KL regularizer E x~p d (x) [D KL (q φ (z|x)kp(z))]
  3. Distance between aggregated posterior and prior D KL (q φ (z)kp(z)).

Analysis of Baselines

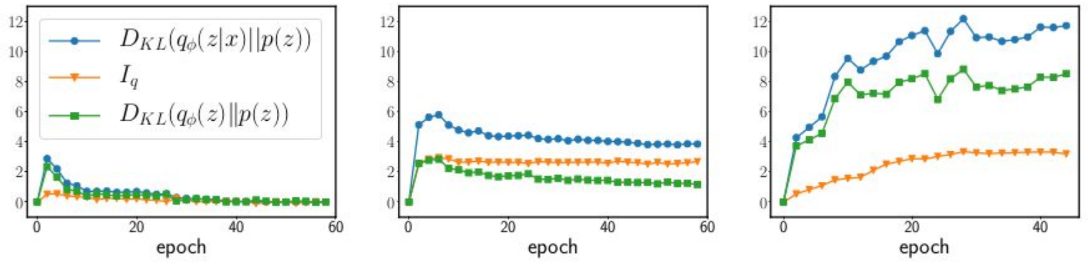


Figure 5: Training behavior on Yelp. **Left:** VAE + annealing. **Middle:** Our method. **Right:** β-VAE ($\beta = 0.2$).

1. 초기에는 모두 증가하는 추새를 보인다.
   Iq in annealing 의 증가가 작다 -> KL regularizer가 크다고 해서 latent variable이 쓰이진 않는다.
2. β-VAE 에서 Iq가 증가하지만, posterior-prior distance DKL (qφ(z)||p(z))도 너무 커지게 된다.
   (pθ(x|z), generator가 qφ(z)에서 샘플된 잠재변수를 사용하기 때문에 항상 작아야된다.)
3. 모델의 prior 보다 z의 가능도가 낮다면 제대로 적합하지 않게 된다.
   => β-VAE 에서 Iq가 클 때 generalize가 잘 안되는 이유.

Analysis of Inner Loop Update

Table 4: Results on Yelp dataset using a fixed budget of inner encoder updates

# Inner Iterations	NLL	KL	MI	AU	Hours
10	357.9	1.1	1.0	3	11.97
30	357.1	3.6	2.5	8	22.31
50	356.9	4.2	2.8	9	29.58
70	357.1	4.4	2.7	10	24.18
convergence	357.0	3.8	2.6	8	21.44

약 30 ~ 100 개의 업데이트가 생성됩니다.
충분한 수의 Inner Loop update가 필요하다는 것을 알지만
성능은 거의 수렴에 가깝게 포화되기 시작하기 때문에 수렴점을 찾는 것이 중요하다.