Adversarial Autoencoders(AAE)

Adversarial Autoencoders

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, Brendan Frey

Abstract

최근 GAN을 확용하여 오토인코더의 잠재 코드 벡터의 집계 사후 확률을 임의의 사전 확률(prior) 분포와 일치시킴으로써 변분 추론을 수행하는 확률적 오토인코더임. VAE는 잠재 변수 z의 분포를 사전 확률 p(z)에 맞추기 위해 KL 발산을 사용히지만 AAE는 GAN의 Discriminator를 사용함.
집계된 사후 분포(aggregated posterior)를 prior과 일치시키는 것은 사전 분포 공간의 어느 부분에서 샘플링하더라도 의미 있는 샘플을 생성할 수 있음을 보장함.
그 결과, AAE의 디코더는 사전 분포를 데이터 분포로 매핑하는 깊은 생성 모델을 학습하게 됨. 다시 말해 prior에서 뽑은 벡터 z를 입력받아 복잡한 실제 데이터 x로 가는 함수를 완벽히 학습함.
AAE는 잠재공간을 정돈하는 능력이 탁월하여 준지도학습, Disentangling등의 작업이 가능함.
VAE나 GAN모델들과 비교했을 때 실제로 우수한 성능을 보임.

정리: AutoEncoder의 잠재공간 z를 GAN의 판별자를 이용해 원하는 분포로 만들어서 잠재 공간을 Dense하게 만들어 prior에서 샘플링할 시에 진짜같은 x가 나오게끔 함.

1. Introduction

데이터의 고차원적인 복잡한 구조(분포)를 찾는 것이 생성모델의 핵심 과제임.
최근까지 제한된 볼츠만 머신(RBM), 심층 신뢰 신경망(DBM), 심층 볼츠만 머신(DMB)과 같은 깊은 생성모델들은 주로 MCMC(Markov Chain Monte Carlo)기반 알고리즘으로 학습됨. 위 모델들은 확률 분포를 '에너지 함수'로 정의하고, 에너지가 낮은 상태(데이터와 유사한 상태)가 나올 확률을 높이는 방식으로 학습함.
이러한 MCMC방법은 로그 우도(log-likelihood)의 gradient를 계산하는데 이는 학습이 진행됨에 따라 점점 더 부정확해짐. 신경망 학습의 핵심은 정확한 기울기(gradient)를 구해 가중치를 업데이트 하는 것인데 이런 에너지 기반 모델은 전체 확률의 합(Partition Function, 분배함수, 정규화 상수 등으로 불림)을 구하는 것이 불가능에 가까워, MCMC로 근사값을 구해야 했음. 학습이 진행되며 근사값의 오차가 커짐으로 학습이 불안정해짐.
이는 MCMC가 Mixing문제 때문임. (Mixing문제란: 샘플링 과정에서 '1' 봉우리에서 샘플을 뽑다가 자연스럽게 건너뛰어 '2' 봉우리에서도 뽑을 수 있어야 하는데, MCMC는 한 봉우리에 갇히면 다른 봉우리로 잘 넘어가지 못하는(갇혀버리는) 문제)
최근 VAE, GAN등이 복잡하고 느린 샘플링 대신, 미분가능한 함수를 통한 역전파를 사용하여 빠르고 안정적으로 학습할 수 있게 됨.

VAE나 중요도 가중 오토인코더(Importance weighted Autoencoders)는 인식 네트워크(Recognition Network, 인코더)를 사용하여 잠재 변수에 대한 사후 분포 $q(z|x)$를 예측함(인코더가 확률분포(μ, σ)를 예측, 단점: 흐릿한 이미지, blurry). GAN은 적재적 학습 절차를 사용하여 역전파를 통해 네트워크의 출력 분포를 직접 형성하며,(가짜를 만드는 생성자와 감별하는 판별자를 경쟁, 단점: 학습이 불안정, Mode Collapse) 생성적 모멘트 매칭 네트워크(generative moment matching networks, GMMN)는 데이터 분포를 학습하기 위해 모멘트 매칭 비용 함수를 사용.(데이터의 통계적 특성(평균, 분산 등 모멘트)를 맞춤)

일반적인 AE는 입력 데이터를 압축, 복원하는 데는 능숙하지만 생성하는 능력은 부족함. 잠재공간이 불규칙하게 학습되기 때문임. AAE는 vanilla autoencoder에 적대적 학습을 추가해 AE를 강력한 생성모델로 전환함.
AAE는 두가지 목표를 가지고 학습함. 첫번째로 재구성 오차, 두번째로 오토인코더의 잠재 표현의 집계된 사후 분포$(q(z))$를 임의의 사전 분포$(\text{prior}, p(x))$와 일치시키는 적대적 손실임.
이 학습 기준이 VAE와 연관되있음. VAE는 KL Divergence라는 수학적 도구를 사용해 분포를 맞추지만, AAE는 판별자라는 신경망을 사용해 분포를 맞춤. 방법은 다르지만 잠재 공간을 정규화(Regularization)한다는 수학적 목적은 동일함.(추후 2.1에서 자세히 증명됨.)
학습이 완료되면 인코더는 데이터 분포를 사전 분포로 변환하는 법을 배우게 되고, 디코더는 부과된 사전 분포를 데이터 분포로 매핑하는 깊은 생성 모델을 학습함.

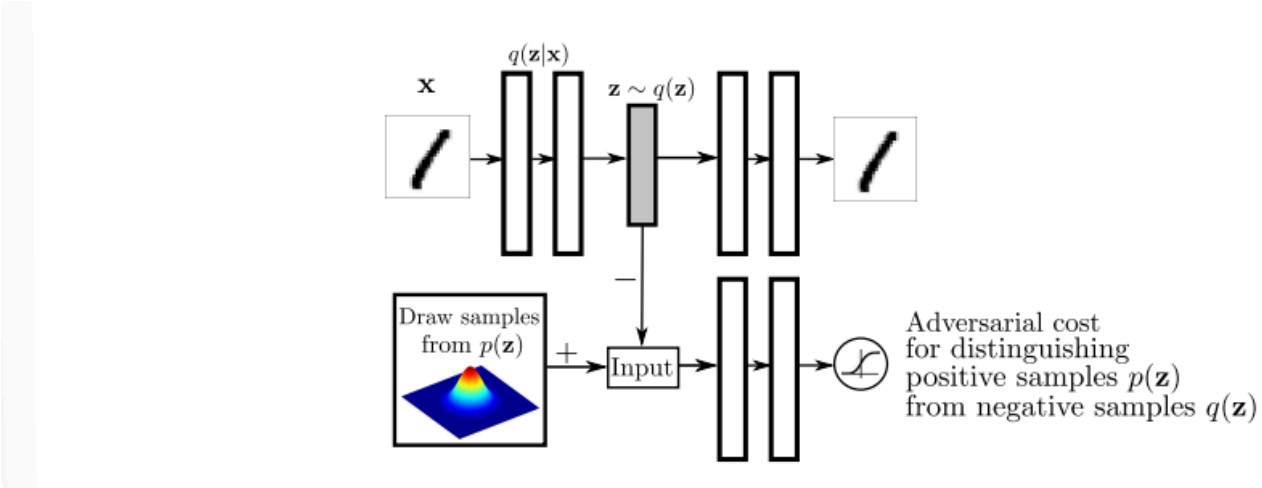


Figure 1: AAE 아키텍처
위쪽은 표준 오토인코더의 아키텍처. 복원된 이미지가 원본 이미지 x와 최대한 똑같아지도록 함. 아래쪽은 적대적 네트워크. 판별자는 받은 샘플 z가 진짜 사전 분포$p(z)$에서 왔는지 가짜 인코더 $q(z)$에서 왔는지 구별하려고 노력함. 생성자(위쪽 행의 인코더)는 판별자를 속이기 위해 자신이 만든 잠재 벡터 z가 마치 p(z)에서 뽑은 것처럼 보이게 만듦.

1.1 Generative Adversarial Networks(GAN)

GAN 프레임워크는 두 신경망(G, D)사이의 minimax 게임을 진행함. G와 D가 서로 반대되는 목적함수를 가짐. 이 경쟁 구도를 통해 잠재공간을 정규화함.
판별자(Discriminator, D(x)) D는 데이터 공간의 어떤 점 x가 생성 모델(Generator G(z))에서 나온

샘플이 아니라, 모델링하려는 실제 데이터 분포에서 나온 샘플일 확률을 계산하는 신경망임.
$D(x) \in [0,1]$: D의 출력은 확률값이므로 0과 1 사이 스칼라임. AAE는 이 x가 아닌 잠재 벡터 z가 된다는 점이 original GAN과의 차이점임.
동시에 생성자는 prior p(z)로부터 샘플링한 z를 데이터 공간(data space)으로 매핑(map)하는 함수(G(z))를 사용. 보통 prior은 $\mathcal{N}(0,1)$을 사용
G는 자신이 생성한 샘플이 실제 데이터 분포에서 유래한 것이라고 믿도록 D를 혼란스럽게(confuse)만들기 위해 학습됨. 이상적인 학습 종료 시점(내시 균형, Nash Equilibrium)에서 D는 x가 어디서 왔는지 구분하지 못해 어떤 입력이 들어오든 확률을 0.5로 출력함.
G는 진짜 데이터가 어떻게 생겼는지 직접 보지 못하고 D가 x를 보고 내린 평가(gradient)를 역전파 받아서 학습함.

$$\min_G \max_D V(D,G) = \min_G \max_D E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p(z)}[\log(1-D(G(z)))]$$

D는 V(D, G) 전체를 최대화하려 함. 진짜 데이터 x에 대해 D(x)=1, 가짜 데이터 G(z)에 대해 D(G(z))=0이 되어야 두 항이 최대가 됨. G는 두번째 항을 최소화하려 함. D(G(z)) ≈ 1이면 $log(0) = -\infty$가 되어 전체 수식 값이 최소화됨.
G와 D는 교대 SGD를 사용하여 학습함. 루프 안에서 고정된 G를 통해 D를 학습한 후 고정된 D를 통해 G를 학습하는 교대 학습 방식을 사용함. AAE에서도 이 방식을 그대로 사용하여 재구성과 정규화를 번갈아 수행함.

2. Adversarial Autoencoders(AAE)

x를 입력, z를 오토인코더의 잠재 벡터라고 할때 $p(z)$를 사전분포, $q(z|x)$를 인코딩 분포, $p(x|z)$를 디코딩 분포, $p_d(x)$를 실제 데이터 분포, $p(x)$를 모델이 생성한 분포라고 정의.

$$q(z) = \int_x q(z|x)p_d(x)dx$$

오토인코더의 인코딩 함수 q(z|x)은 잠재벡터에 대한 집계된 사후 분포 q(z)로 정의함. q(z|x)는 데이터 하나(x)가 들어왔을 때 잠재 벡터(z)가 어디에 찍히는지에 대한 개별 분포. 이를 모든 x에 대해 적분할시 전체 데이터셋을 다 인코더에 통과시켰을 때 만들어지는 z들의 총합 분포임.
AAE는 q(z)를 p(z)와 일치시킴으로써 정규화된 오토인코더임. 이를 위해 figure 1과 같이 오토인코더의 잠재 벡터(hidden code vector)위에 적대적 네트워크가 부착됨.
이때 적대적 네트워크의 생성자 G는 오토인코더의 인코더 q(z|x)임. 인코더는 고차원 데이터를 압축하는 동시에, 압축한 z가 p(z)에서 샘플링 했을법한 z를 생성해야 함.
인코더의 목표는 $q(z|x) \approx p(z)$ 임.
적대적 네트워크와 오토인코더는 각 미니 배치마다 실행되는 재구성과 정규화를 통해 공동으로 학습됨.
그 다음 적대적 네트워크는 생성자(인코더)를 업데이트하여 D를 혼란스럽게 만듦. 학습 절차가 완료되면, 오토인코더의 디코더는 $p(z)$를 데이터 분포로 매핑하는 생성 모델을 정의하게 됨.
AAE의 인코더($q(z|x)$)를 정의하는 세 가지 방법을 제시함.

1. 결정론적 z = f(x). 입력 x가 같으면 항상 똑같은 z가 나옴. -> 구현이 쉽고 단순(Vanilla AE)
2. 가우시안 사후 분포 $z \sim \mathcal{N}(\mu(x), \sigma(x))$, 인코더가 평균과 분산을 예측함. 재매개변수화 트릭(Re-parameterization trick)을 사용해야 역전파가 가능함. -> 데이터의 불확실성을 모델링할 수 있음.
3. Universal approximator posterior(보편 근사자): $z = f(x, \eta)$. 입력 x와 랜덤 노이즈 η를 같이 넣음. q(z|x)가 가우시안일 필요 없이 어떤 모양이든 될 수 있는 유연함을 가짐.

$$q(z|x) = \int_{\eta} q(z|x,\eta)p_{\eta}(\eta)d\eta \Rightarrow q(z) = \int_{x}\int_{\eta} q(z|x,\eta)p_{d}(x)p_{\eta}(\eta)d\eta dx$$

데이터(x)의 확률과 노이즈(η)의 확률이 결합되어 z의 분포를 만든다는 것임.
VAE는 수학적으로 미분하기 위해 반드시 잠재 변수가 가우시안이라고 가정해야 했지만 AAE는 노이즈를 입력으로 받아 신경망을 통과시키므로, 잠재 분포 설정이 유연함.
VAE는 반드시 인코더가 확률적이어야 학습이 되는데, AAE는 결정론적 함수만 써도 GAN이 알아서 분포를 맞춰주기 때문에 잘 작동함.

2.1 Relationship to Variational Autoencoders

AAE는 VAE와 본질적으로 유사함. VAE는 잠재벡터의 사전 분포를 부과하기 위해 KL 발산항을 사용하였고, AAE는 잠재 벡터의 집계된 사후 분포를 사전 분포와 일치시키는 적대적 학습절차를 도입. 결국 목적($q(z) \approx p(z)$)은 같음.

$$E_{x\sim p_d(x)}[-log\, p(x)] < E_x[E_{q(z|x)}[-log(p(x|z))]] + E_x[KL(q(z|x)||p(z))]$$

$$= E_x[E_{q(z|x)}[-log\, p(x|z)]] - E_x[H(q(z|x))] + E_{q(z)}[-log\, p(z)]$$

$$= \text{Reconstruction} - \text{Entropy} + \text{CrossEntropy}(q(z), p(z))$$

VAE의 수식을 세 부분으로 쪼갬.
- Reconstruction(재구성): 입력 x를 z로 압축했다가 다시 x로 복원했을 때의 오차.
- Entropy: '무질서도' 또는 '불확실성'임. 앞에 마이너스가 있으므로, 이 식을 최소화한다는 것은 엔트로피를 최대화한다는 뜻임. 즉, 인코더가 z를 한 점으로 찍지 말고, 가능한 넓게 퍼뜨리라(분산 σ를 키워라)는 압박.
- CrossEntropy: q(z)와 p(z)사이의 차이. 이를 최소화하는건 $q(z) \approx p(z)$의 의미임.
- 정규화 항(KL Divergence)가 엔트로피항과 교차 엔트로피 항으로 분해됨.

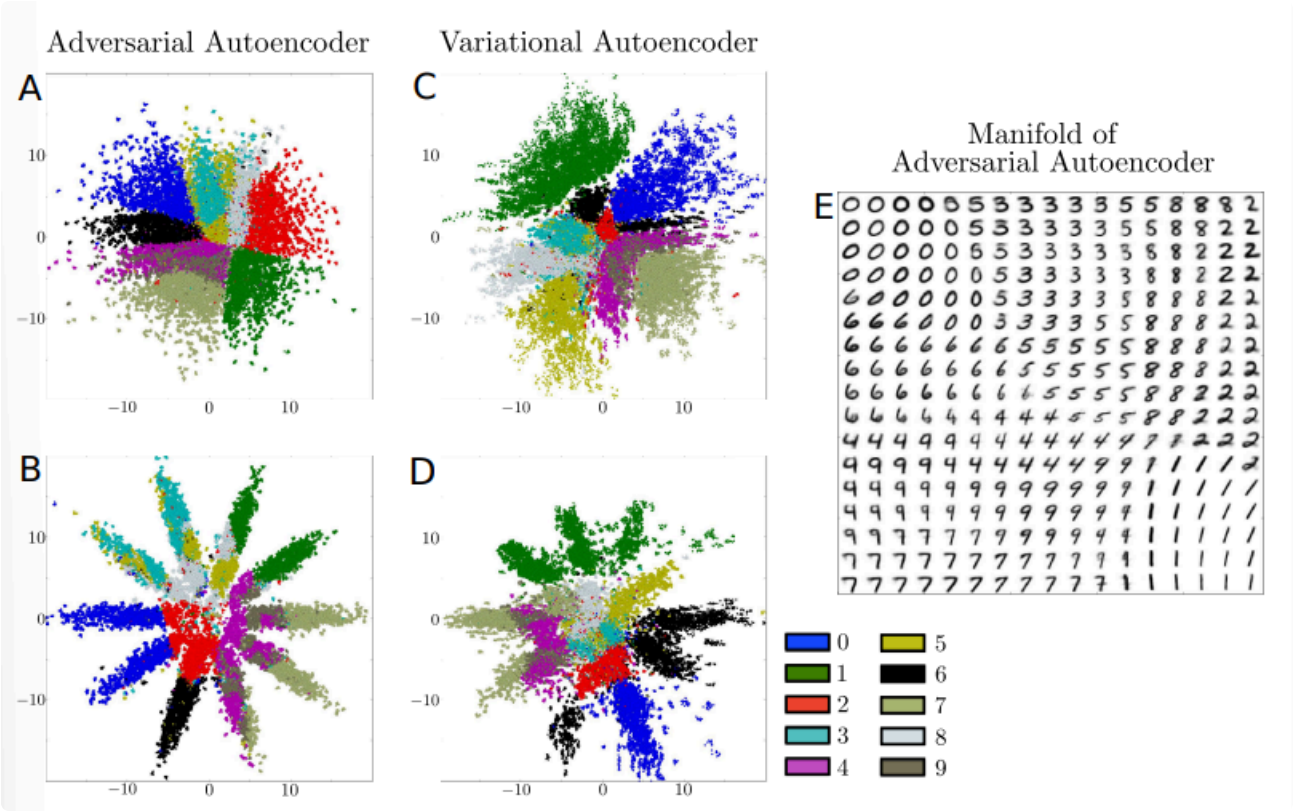


Figure 2a는 AAE의 잠재벡터공간의 학습된 다양체는 구멍이 없음. 반면 VAE의 잠재벡터공간에서는 중간중간 빈 공간이 보이고 가장자리가 흐릿함. figure 2e는 2a의 잠재공간에서 각 차원의 **백분위수를 따라 균일하게 샘플링**한 플롯임. 이는 단순히 z값을 일정한 간격으로 자른 것이 아니라 확률(누적 분포 함수, CDF)을 기준으로 일정한 간격으로 잘랐다는 뜻임. 이로써 모든 종류의 데이터를 발생 확률에 비례해서 공평하게 시각화함.

VAE의 세 개의 항을 봤을 때 정규화 항이 없다면 단순히 입력을 재구성하는 표준 오토인코더일 뿐임. 생성 모델이 아니라 단순 압축기.
정규화 항이 존재함으로써 VAE는 p(z)와 호환되는(compatible, 잠재 벡터들이 p(z)의 형태를 갖춘다는 뜻) 잠재 표현을 학습.
두번째 항($-E[H(q(z|x))]$)에서 엔트로피를 최대화해야 함. 다시 말해, 불확실성이 크다는 것이고, 가우시안 분포에서 불확실성이 크다는 것은 분산이 넓게 퍼진다는 뜻임. 즉, VAE는 하나의 점으로 수렴하는 것을 막고, 데이터를 넓게 퍼뜨리려는 성질을 가짐.
교차 엔트로피 항은 q(z)가 p(z)의 모드를 선택하도록 장려하는데 이는 p(z)의 전체 분포를 꽉 채우기보다는 확률이 높은 중심부만 맞추려는 경향이 있음.
반면, AAE에서 엔트로피와 교차엔트로피 항을 적대적 학습 절차로 대체하여 q(z)가 p(z)의 전체 분포와 일치하도록 장려함. AAE가 VAE보다 **더 Dense한 잠재 공간**을 만들 수 있는지에 대한 이론적 근거를 제공함.
Figure 2를 통해 AAE와 VAE가 prior대로 잠재공간을 잘 빚어내는지 비교함.
prior로 2D 가우시안이 부과되었을 때, AAE는 각 클래스가 서로 뭉개지지 않고 경계선이 선명함. 중간에 빈 공간이 없음. 그 결과 Figure 2e(부드럽게 변하는 숫자 이미지들)를 도출할 수 있음.
같은 prior에 대해 VAE는 대략적으로 일치함. 그러나 잠재 공간에 구멍이 존재함.

사전분포를 10개 혼합 가우시안으로 설정했을때도 VAE에 비해 AAE의 잠재공간이 깔끔하게 분리되어 있고 구멍도 적음.

VAE의 한계는 KL 발산 항을 계산하고 미분하려면 p(z)가 수학적으로 명확하게 정의된 식이어야 함. 따라서 prior을 단순한 분포밖에 사용할 수 없음. 그에 반해 AAE는 사전 분포로부터 샘플링만 할 수 있으면 됨. 수식은 몰라도 상관없음. 2.3에서 그 점을 증명.

2.2 Relationship to GANs and GMMNs

일반적인 GAN은 생성자가 곧바로 고차원 이미지를 만들고, D는 고차원 이미지를 보고 진짜/가짜를 판별함.
AAE는 데이터 분포를 포착하기 위해 AE학습에 의존함. 이미지를 그리는 작업은 AE한테 맡김.
고차원에서 D 학습이 불안정한 원인을 해결하기 위해 잠재 공간으로 끌고 옴.

> Q. 테스트 우도(test-likelihood)란?
> A. 수학적으로 우도(Likelihood)는 모델이 파라미터 θ를 가질 때, 관측된 데이터 x가 등장할 확률 밀도 $p_\theta(x)$를 뜻함. 테스트 데이터셋 $X_{test} = x_1, x_2, \ldots, x_n$이라고 하고, 모델의 확률 분포를 p_{model}이라고 할때 테스트 로그 우도는 $\sum_{i=1}^{n} \log P_{model}(x_i)$로 정의됨. 즉, 테스트 데이터에 있는 진짜 이미지들(x_i)이 우리 모델에서 생성될 확률이 얼마나 높은지를 다 더한 값임.

Generative moment matching networks(GMMN)는 신경망 출력 레이어의 분포를 형성하기 위해 MMD(Maximum Mean Discrepancy, 최대 평균 불일치) 목표를 사용함. GAN처럼 D(신경망)를 쓰는 게 아니라, 통계적 수치(평균, 분산 등 모멘트)를 비교해서 분포를 맞추는 방식임.
GMMN은 더 나은 우도 결과를 얻기 위해 사전 훈련된 dropout AE와 결합될 수 있음.
사전학습된 AE의 고정된 잠재공간에다가 억지로 분포를 맞추는 GMMN+AE에 비해 AAE은 AE를 처음부터 학습시키면서 정규화를 함. 결과적으로 압축과 최적의 잠재 공간이 형성됨.
섹션 3에서 AAE의 학습 결과(테스트 우도)가 여러 데이터셋에서 GMMN 및 GMMN_AE를 능가함을 보임.

2.3 Incorporating Label Information in the Adversarial Regularization

데이터에 레이블이 있는 경우에 이를 활용하여 잠재 공간의 분포를 더 잘 형성할 수 있음. 가령, Figure 2b의 10개의 2D 가우시안 혼합 분포에 각 모드가 단일 레이블(숫자)로 강제할 수 있음.

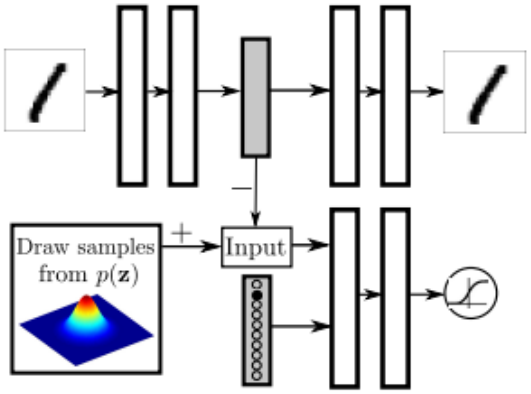


Figure 3은 이러한 준지도(semi-supervised) 접근 방식의 훈련 절차를 보여줌. 분포의 모드와 레이블을 연관시키기 위해 D의 입력 z와 레이블(원 핫 벡터)을 추가함. 판별 기준을 레이블에 따라 바꿈으로써, 인코더가 해당 레이블의 이미지를 특정 위치로 보내도록 강제함.

> Q. prior에서 샘플링한 z에는 레이블이 없지 않은가? 이 데이터에 대해 모델(판별자)이 어떤 mode로 가라고 지시하는가?
> A. 특정 Mode로 가라고 지시하지 않는 대신, 전체 분포 중 어딘가에만 속하면 된다고 지시함. 동시에 별도의 분류기 q(y|x)가 "이 모양은 1번 Mode에 가깝네"라고 확률적으로 학습을 보조함.

레이블이 없는 경우나 추가 클래스(Mnist에서 11번째 클래스의 경우)가 존재하는 경우 이때는 아무 모드로 보냄.
여기서 '보낸다'라는건 인코더가 능동적으로 보내는게 아닌 전체 혼합 분포 안으로 매핑함.
학습 과정에서 보면 D가 True 데이터(p(z))를 받을 때 레이블이 있든 없든 1이라고 학습하고, Fake 데이터(q(z))를 받을 때 학습 데이터의 레이블을 D에게 제공함.
Positive 샘플: 직접 레이블(y)을 고른 후 해당 가우시안에서 z를 샘플링.
Negative 샘플: 이미지 x의 실제 레이블 y를 가져옴.

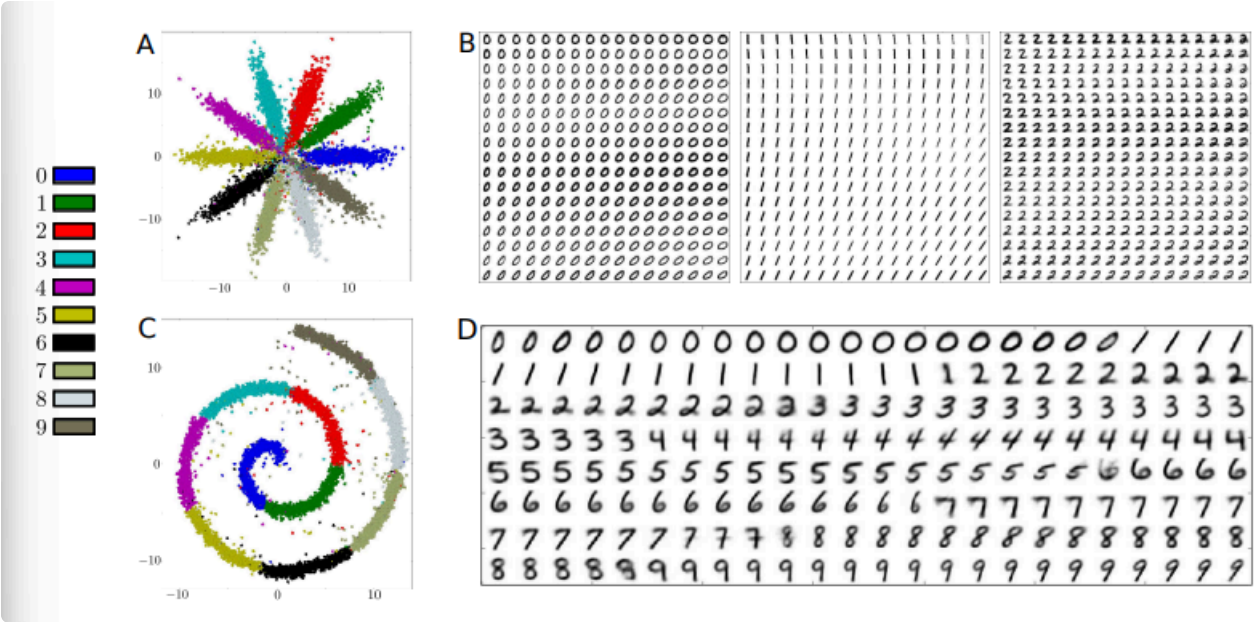


Figure 4a는 Mnist 데이터셋에서 레이블이 있는 데이터 1만개, 레이블이 없는 데이터 4만개에 대해 훈련된 10개의 2D 가우시안 혼합 사전 분포를 가진 AAE의 잠재 표현임.

Figure 4b는 위 사전 분포의 처음 세개의 mode를 시각화함. 또한 클러스터 내 좌표에서 추상적인 스타일(기울기) 개념을 학습함을 보임.
Figure 4c, 4d 방법은 모수적 형태가 없는 임의의 분포로 확장될 수 있으며 MNIST데이터셋을 스위스 롤에 매핑함으로써 입증함. 4c는 잠재 공간, 4d는 스위스 롤 축을 따라가며 생성된 이미지임.

> Q. 모수적 형태가 없는 스위스 롤은 분포를 모르는데 어떻게 샘플링하는가?
> A. 확률 밀도 함수의 수식(p(z) = ...)은 몰라도 되지만, 그 모양을 그리는 절차(Algorithm with python)만 알면 샘플링이 가능함. 이 과정에서 특정 점이 나올 확률 밀도가 얼마인지 확인하는 수식은 필요 없음. 그냥 알고리즘대로 생성된 점들이 스위스 롤 모양을 이루기만 하면, 판별자는 그 모양을 학습함.

3. Likelihood Analysis of Adversarial Autoencoders

이 섹션에서는 수치적(정량적)으로 얼마나 성능이 좋은지를 증명함.
GAN에서 언급된 평가 절차 사용, MNIST(8D 잠재공간)와 TFD(15차원 잠재공간) 테스트 이미지 생성하는 모델의 likelihood를 비교.

(a) MNIST samples (8-D Gaussian)

(b) TFD samples (15-D Gaussian)

모델이 과적합되었는지 확인하기 위해 Figure 5의 마지막 열을 끝에서 두번째 열에 있는 샘플링된 샘플과 유클리드 거리상 가장 가까운 이웃을 보여줌. 만약 두 열의 샘플이 거의 일치했다면 과적합이지만 서로 다름.(나머지 열도 샘플임.)
p(x)라는 완벽한 수식을 모르기 때문에, 테스트 이미지 x를 넣었을 때 확률값이 툭 튀어나오지 않음. 따라서 실제 로그 우도의 하한을 계산. 모델에서 생성된 1만개 샘플에 Gaussian Parzen Window(kernel density estimator, KDE)를 fit시키고, 이 분포 하에서 테스트 데이터의 우도를 계산함.

> Q. 가우시안 파르젠 윈도우란?
> A. 히스토그램을 아주 부드럽게 그린 것. 샘플 1만개 각 점 위에 각각 작은 가우시안 분포를 씌움. 그 언덕들을 다 더하면 전체적인 확률 지형도가 생김. 이 지형도 위에서 진짜 테스트 이미지들

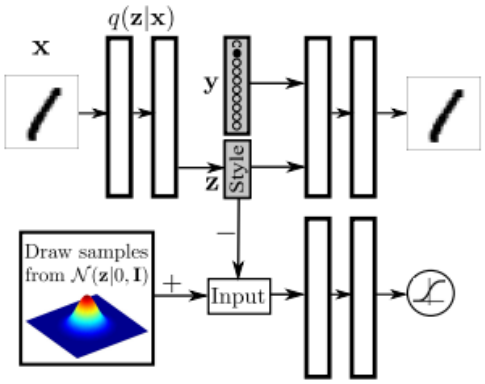
> 의 위치(확률 높이)를 잼.
> 현재는 FID, IS, Precision and Recall 등을 사용

	MNIST (10K)	MNIST (10M)	TFD (10K)	TFD (10M)
DBN [Hinton et al., 2006]	138 ± 2	-	1909 ± 66	-
Stacked CAE [Bengio et al., 2013]	121 ± 1.6	-	2110 ± 50	-
Deep GSN [Bengio et al., 2014]	214 ± 1.1	-	1890 ± 29	-
GAN [Goodfellow et al., 2014]	225 ± 2	386	2057 ± 26	-
GMMN + AE [Li et al., 2015]	282 ± 2	-	2204 ± 20	-
Adversarial Autoencoder	$\mathbf{340 \pm 2}$	**427**	$\mathbf{2252 \pm 16}$	**2522**

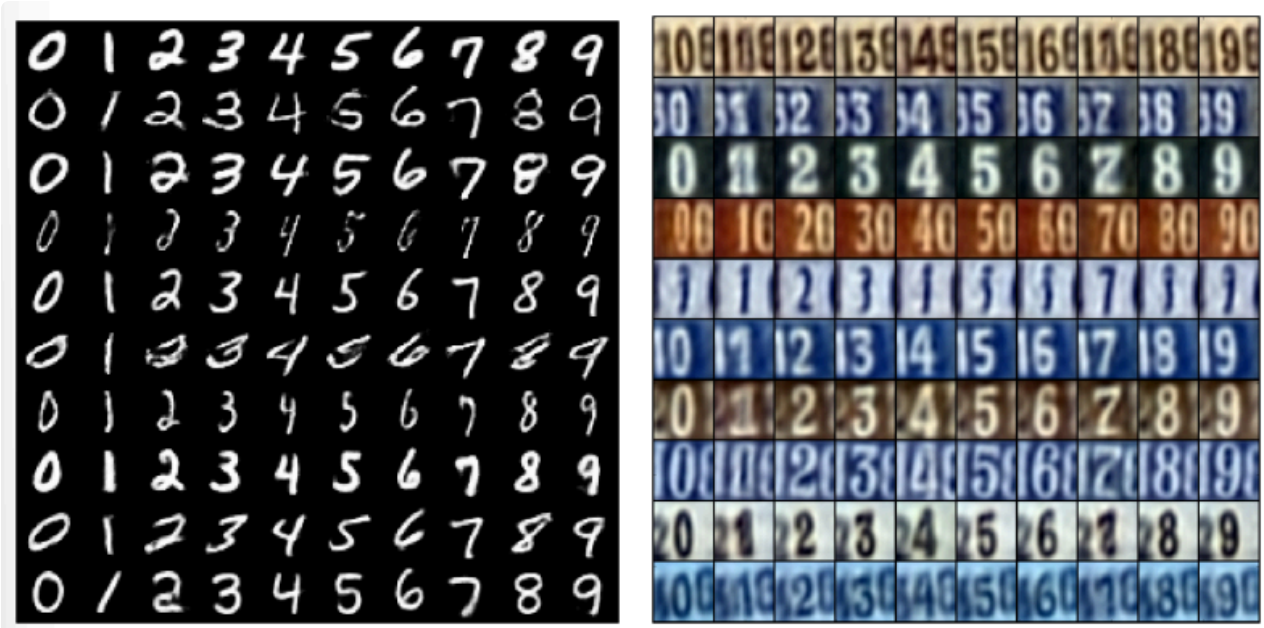
Table1은 MNIST와 TFD에 대한 AAE의 로그 우도를 DBN, GAN, GMMN+AE 등 많은 최신 방법들과 비교. 이를 통해 AAE가 상대적으로 우월한 로그 우도를 달성함.(데이터 분포를 가장 잘 학습)
하지만 파르젠 윈도우 방식은 고차원 공간에서 정확도가 떨어짐에도 불구하고, 더 나은 대안이 없었기에 쓰임. 이를 명시함.

4. Supervised Adversarial Autoencoders

AAE를 이용해 내용(label)과 스타일(style)을 분리하는 Disentanglerment(얽힘 풀기)를 다룸.
지도학습 상황에서 AAE로 레이블과 스타일을 어떻게 분리하는지 보여줌. 그 다음 5장에서 준지도 설정으로 확장.



이전 디코더는 잠재벡터 z만 받았다면 현재 아키텍처에서는 레이블 정보y도 추가로 받음. 이를 통해 디코더는 y를 통해 이 이미지가 어떤 레이블인지 알기 때문에 인코더가 굳이 z안에 레이블 정보를 넣을 필요가 없음. -> z는 순수한 스타일 변수가 됨.



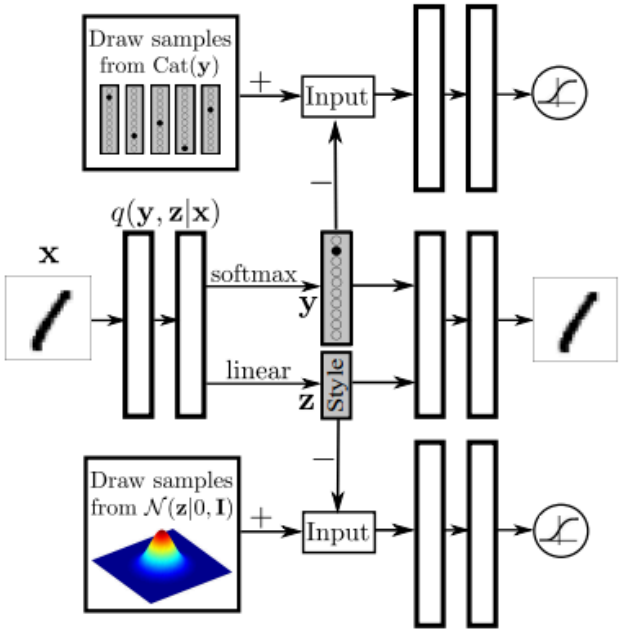
(a) MNIST (b) SVHN

Figure 7a는 잠재 공간이 15차원 가우시안으로 강제된 네트워크의 결과를 보임. 각 행에서 숫자는 바뀌지만 글씨체(굵기, 기울기, 날림 정도)는 똑같음. 이는 z가 스타일을 담당하고, y가 숫자(Label)를 담당하는 것을 증명함.
Figure 7b는 거리 주택 번호(SVHN) 데이터셋에 동일한 실험을 적용한 것을 보여줌. 가로줄을 보면 숫자는 바뀌지만 양옆의 숫자와 배경 색깔은 유지됨.
Fugire 7 실험에서 각 행이 모두 같은 모델에서 샘플링된 샘플임.

5. Semi-Supervised Adversarial Autoencoders

데이터가 범주형 분포에서 나오는 이산 클래스 변수 y와 가우시안 분포에서 나오는 연속 잠재 변수 z에 의해 생성된다고 가정.

$$p(y) = Cat(y), p(z) = \mathcal{N}(z|0, I)$$



인코더 q(y, z|x)를 사용하여 이산 클래스 변수 y와 연속 잠재 변수 z를 모두 예측하도록 Figure 6의 아키텍처를 변경(Figure 8). 그러면 디코더는 이미지를 재구성하기 위해 원-핫 벡터로서 클래스 레이블과 연속 잠재 코드 z를 둘 다 활용함.

두개의 별도 적대적 네트워크

첫번째 적대적 네트워크: 레이블 표현(y)에 범주형 분포를 부과.

두번째 적대적 네트워크: 스타일 표현(z)에 가우시안 분포를 부과.

여기서 y에 대한 적대적 학습의 의미: 판별자는 이 y가 진짜 원-핫 벡터인지 검사(소프트맥스 출력이 아닌 제일 높은 확률값에 1 부여). 이렇게 하면 레이블이 없는 데이터에 대해서도 모델이 스스로 레이블을 분류하는 능력을 키우게 됨.

두 적대적 네트워크는 SGD를 통해 재구성(non label x를 써서 스스로 분류와 z를 뽑아 다시 복원), 정규화(y가 범주형 분포를 따르도록, z는 가우시안을 따르도록 판별자와 경쟁), 준지도 분류(레이블이 있는 데이터를 사용하여 인코더의 분류 능력을 직접 학습) 세 단계로 훈련됨.

	MNIST (100)	MNIST (1000)	MNIST (All)	SVHN (1000)
NN Baseline	25.80	8.73	1.25	47.50
VAE (M1) + TSVM	11.82 (±0.25)	4.24 (±0.07)	-	55.33 (±0.11)
VAE (M2)	11.97 (±1.71)	3.60 (±0.56)	-	-
VAE (M1 + M2)	3.33 (±0.14)	2.40 (±0.02)	0.96	36.02 (±0.10)
VAT	2.33	1.36	0.64 (±0.04)	24.63
CatGAN	1.91 (±0.1)	1.73 (±0.18)	0.91	-
Ladder Networks	1.06 (±0.37)	0.84 (±0.08)	0.57 (±0.02)	-
ADGM	0.96 (±0.02)	-	-	16.61 (±0.24)
Adversarial Autoencoders	1.90 (±0.10)	1.60 (±0.08)	0.85 (±0.02)	17.70 (±0.30)

Table 2를 통해 AAE모델의 준지도학습이 우수하다는 것을 보여줌. 또한 모든 AAE모델은 end to end로 훈련되는 반면, 준지도 VAE모델은 단계별로 훈련되어야 함.

6. Unsupervised Clustering with Adversarial Autoencoders

이전 섹션에서 제한된 레이블 정보를 통해 준지도 표현학습이 가능하다는 것을 증명함. 이번엔 레이블이 없는 비지도 task에도 AAE가 강력한지 보여줌.

Figure 8에서 y에 대한 정답을 알려주던 과정을 제거하여 스스로 분류 기준을 세우게끔 유도. 추론 네트워크 q(y|x)가 데이터를 클러스터링하고자 하는 카테고리 수 만큼의 차원을 가진 원-핫 벡터를 예측함.(ex. mnist니까 대략 10~16개 정도로 잠재공간을 정의)

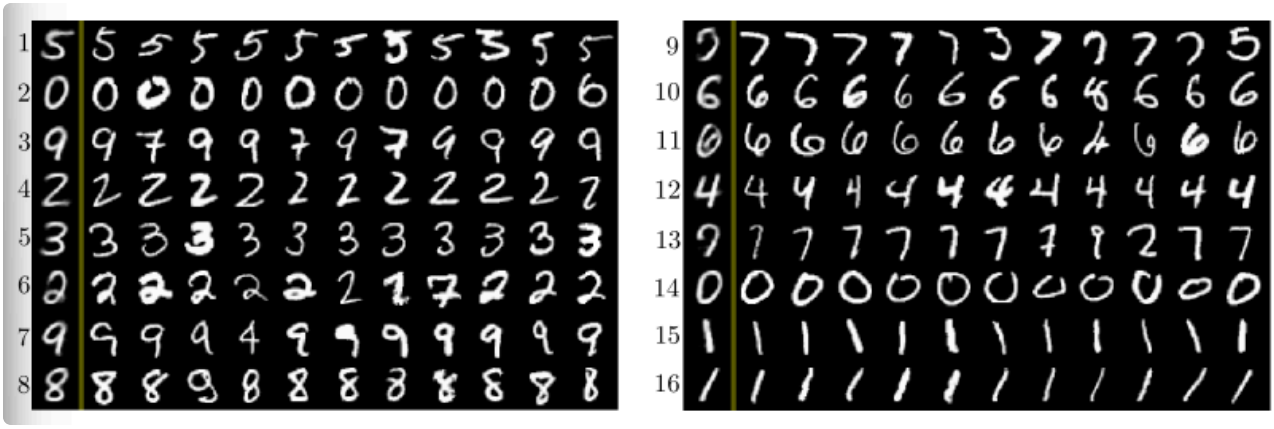


Figure 9는 클러스터 수가 16일 때 MNIST에 대한 AAE의 비지도 클러스터링 성능을 보임. Cluster Head로 스타일 변수 z를 0으로 고정하고 레이블 변수 y를 원-핫 벡터 중 하나로 설정하여 생성하고 나머지는 해당 카테고리로 분류된 무작위 테스트 이미지임.
AAE가 일부 이산적인 스타일을 클래스 레이블로 포착함. 15,16번 클러스터를 보면 기울기가 다른데 이런 미세한 구조적 차이까지 스스로 감지한다는 것을 의미함.

	MNIST (Unsupervised)
CatGAN [Springenberg, 2015](20 clusters)	9.70
Adversarial Autoencoder (16 clusters)	9.55 ($\pm$2.05)
Adversarial Autoencoder (30 clusters)	4.10 ($\pm$1.13)

훈련이 끝난 후 각 클러스터 i에 대해 $q(y_i|x_n)$을 최대화하는(가장 확신하는) 이미지 x_n을 찾고, 그 x_n의 실제 레이블을 해당 클러스터 i의 모든 포인트에 할당하여 에러율을 계산한 결과 클러스터 수가 늘어날수록 분류율이 향상됨을 관찰.(ex. 클러스터를 30개쯤 넉넉하게 주면, "너는 기울어진 1 방해, 나는 똑바른 1 방 할게"라며 평화롭게 나눠지고, 나중에 우리가 "둘 다 1이야"라고 합쳐주면 되기 때문에 정확도가 올라감.)

7. Dimensionality Reduction with Adversarial Autoencoders

이전에 비모수적 차원 축소 기법들의 단점은 새로운 데이터 포인트의 임베딩(인코더의 부재)을 사용할 수 없음. 일반 오토인코더는 연속성이 깨지지만 AAE의 판별자가 분포를 강제하므로 연속성을 부여함.

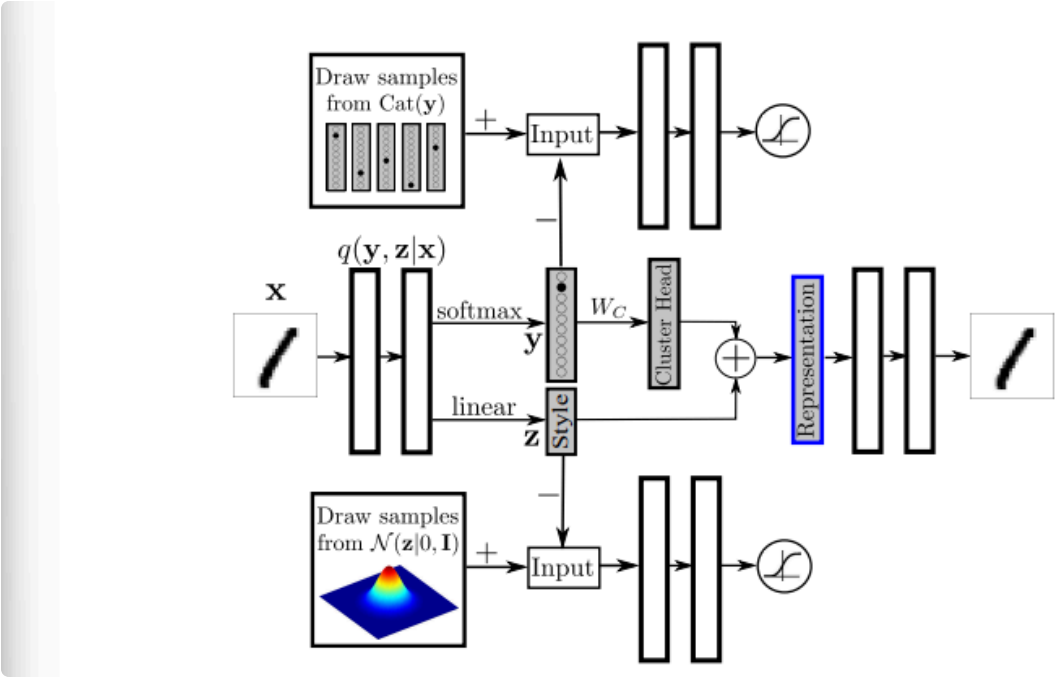
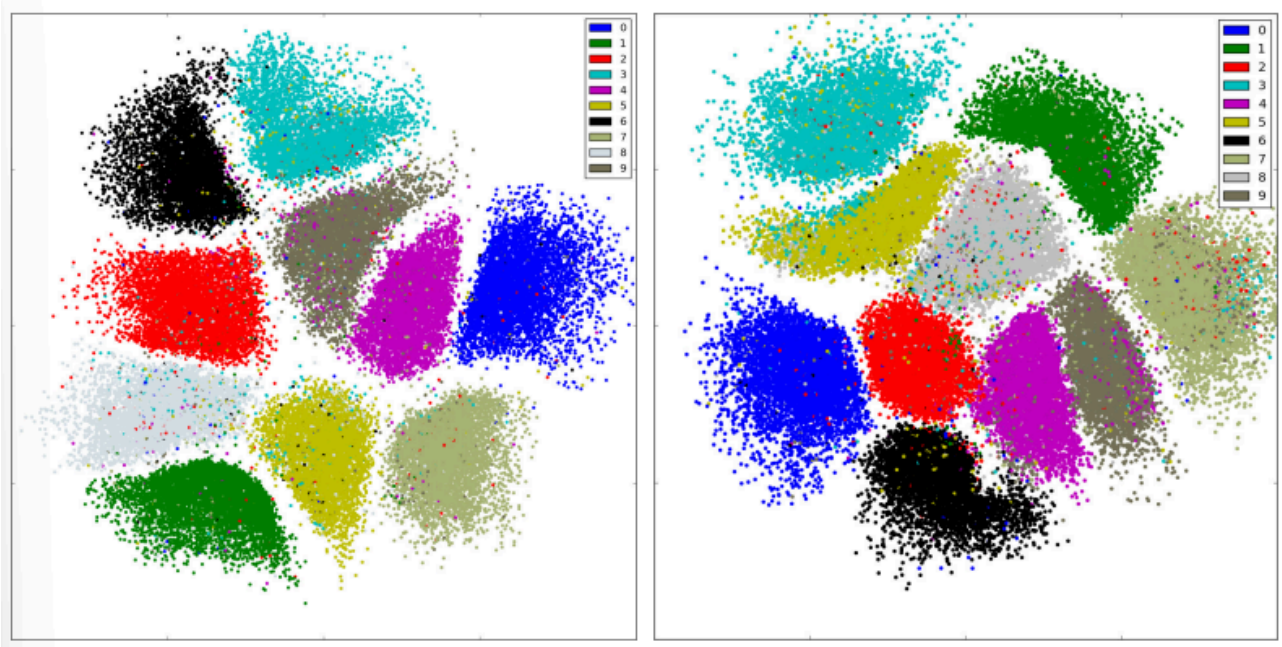
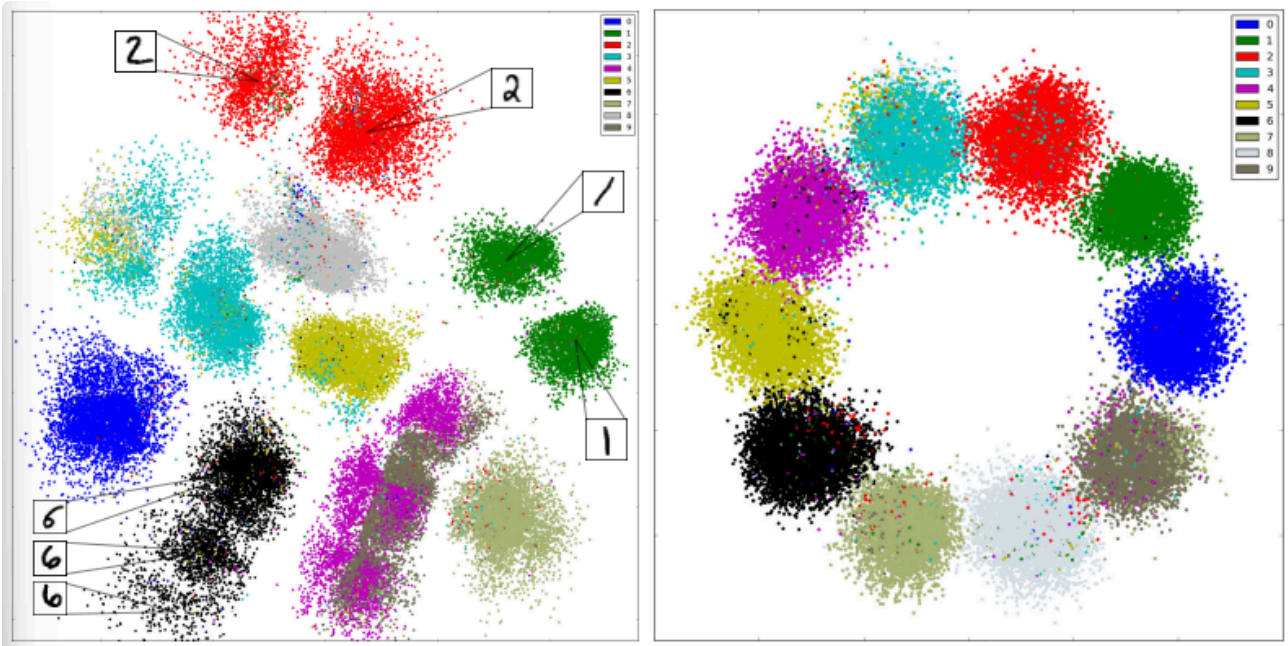


Figure 10을 보면 m개의 레이블 정보가 주어진 데이터셋에서 잠재공간을 2, 3인 n차원으로 축소한다고 가정할 때, 최종 표현은 n차원 클러스터 헤드 표현과 n차원 스타일 표현을 더함으로써 얻어짐.
(최종 좌표 z_{style} = 클러스터 중심(Head) + 스타일 변동)
각 숫자가 2차원 지도상의 어디에 위치할 지 결정하는 좌표값(W_C) 자체도 학습 대상임.
두 클러스터 헤드 사이의 유클리드 거리에 패널티를 주는 비용함수를 도입하여 각 클러스터의 중심들(W_C)이 서로 붙어있지 않고, 적당히 떨어지기를 기대.



(a) 2D representation with 1000 labels (4.20% error) (b) 2D representation with 100 labels (6.08% error)

Figure 11(a,b)는 각각 1,000, 100개의 레이블을 사용(준지도)하여 mnist데이터셋(m=10)을 2차원으로 축소함. 분류 오차는 고차원마늠 좋지 않으며, 각 클러스터의 분포가 완벽한 가우시안은 아님.



(c) 2D representation learnt in an unsupervised fashion with 20 clusters (13.95% error)

(d) 10D representation with 100 labels projected onto the 2D space (3.90% error)

Figure 11c는 m=20으로 설정했을 때 네트워크가 스타일까지 고려하여 클러스터를 할당.
Figure 11d는 n=10으로 차원을 넓힘. 10D 가우시안 분포로 학습후 시각화를 위해 2D 원 위에 균일하게 배치되도록 2D 공간으로 매핑.

8. Conclusion

GAN 프레임워크를 확률적 오토인코더의 이산(카테고리, 클래스) 및 연속(가우시안, 스타일) 잠재변수 모두에 대한 변분 추론 알고리즘으로 사용할 것을 제안함. 기존의 복잡한 수식 KL 발산 대신 적대적 학습을 이용해 확률 분포를 근사하는 방법 제시.
AAE는 데이터의 진짜 분포를 수학적으로 잘 모델링하며 경쟁력있는 테스트 우도를 달성함.
준지도 task, disentangling, 비지도 클러스터링, 차원 축소 및 시각화에 대한 응용 사례를 보임.

정리

AE의 잠재 공간을 복잡한 KL Devergence 대신 GAN의 프레임워크를 도입하여 정규화할 수 있음.
prior를 유연하게 설정 가능함.
분리 능력도 탁월함.