Knowledge Extraction with No Observable Data

U Kang*
Seoul National University

NeurIPS 2019

Cite : 6

2020.07.23
임진혁

0 Index

1. Introduction

2. Related work

3. Proposed Approach

4. Experiments

1 Introduction: How can we distill the knowledge without any observable data?

Abstract

knowledge distillation은 파라미터가 크고 deep한 모델 또는 ensemble 모델의 뛰어난 성능(지식)을 다른 모델에게 전달하는 방법론으로써정보 함유량이 높게 만든 transfer data로 학습하기 때문에 데이터가 적거나 다른 모델(Student)의 용량이 작을 경우 더욱 효과적이다.

- **지금까지 줄곧 배워온 KD의 기본 구조를 생각해보자**

  - 높은 성능을 가진 네트워크 T가 존재

  - 훈련데이터 x에 대한 T의 예측값을 Ground Truth 대신 네트워크 S의 훈련 데이터로 사용

즉, KD에서 데이터는 필수적이다. 그 데이터를 통해서 T의 Knowledge가 S에게 전파되기 때문이다.

물론 기존 연구들 중에서 매우 소수의 샘플로만 지식전파를 하는 메타학습 또는 few shot관련 kd 방법론이 제안되었지만
본 연구는 아예 train-data를 전혀 알 수 없고 관측할 수 없는 상황에서 KD를 진행하기 위한 방법론
KEGNET (Knowledge Extraction with Generative Networks)를 제시한다.

1 Introduction: How can we distill the knowledge without any observable data?

KD 기본적인 한계 :

데이터가 Knowledge Transfer의 수단(통로)이므로
데이터가 충분하지 않거나 아예 없는 경우 전파가 불가능하다.
Q: 데이터가 아예 없는 경우가 뭐가 있을까?

Q: 아무 데이터나 T의 input으로 주고 그 예측값을(결과값) 사용하면 안되는가?

1 Introduction: How can we distill the knowledge without any observable data?

KD 기본적인 한계 :

데이터가 Knowledge Transfer의 수단(통로)이므로
데이터가 충분하지 않거나 아예 없는 경우 전파가 불가능하다.
Q: 데이터가 아예 없는 경우가 뭐가 있을까?
--> Privacy or Confidential Issue

Q: 아무 데이터나 T의 input으로 주고 그 예측값을(결과값) 사용하면 안되는가?
→ 무작위 데이터를 주어서 생성할 수도 있겠지만, 그러한 데이터는 실제 X랑 전혀 다른 분포를 가지고 있어 오히려 성능이 저하될 수 있다.
→ manifold 가정

KEGNET (Knowledge Extraction with Generative Networks)

지식을 증류하기에 앞서 T의 지식을 먼저 추출해야 하기 대문에 지식 추출이라고 이름을 지었다.

기본적인 구조는 T에서 추출한 지식으로 만든 데이터가 (원래 관측 데이터)를 대체한다는 점에서 GAN과 비슷하고 디코더 구조가 존재한다는 점에서 오토인코더랑 비슷

But Gan -> need train data
Decoder -> make low dimension represntation

2 Related work

(1) [Semisupervised knowledge transfer for deep learning from private training data][28]

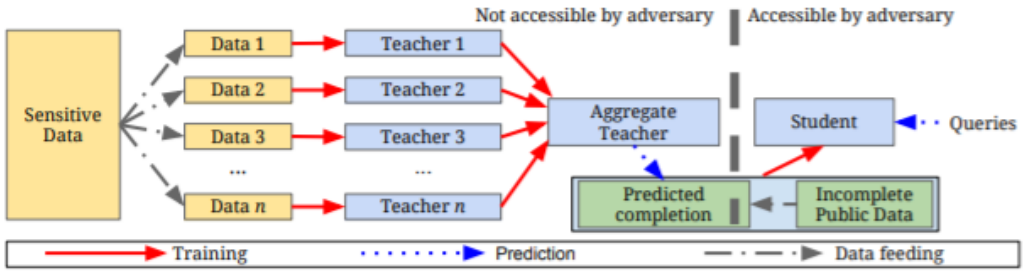
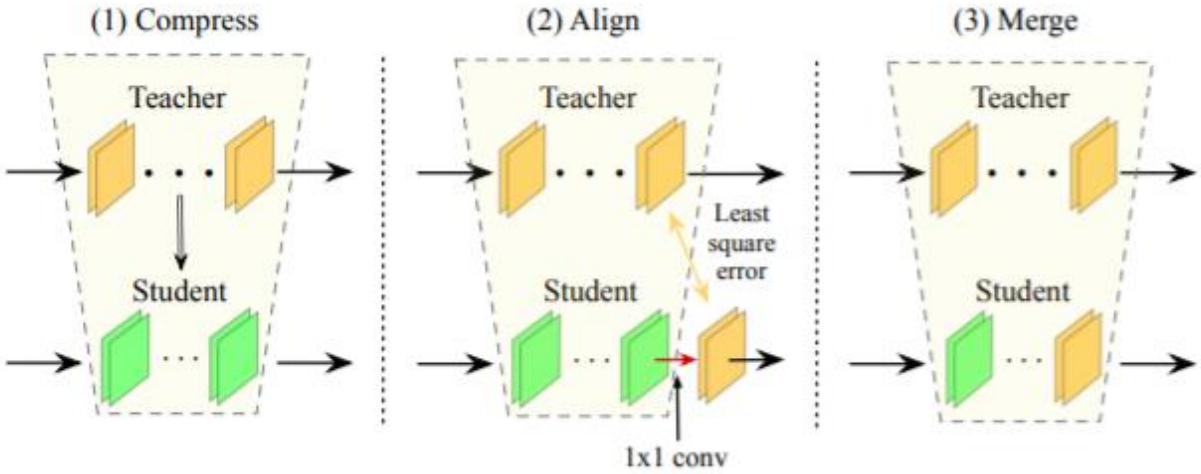


Figure 1: Overview of the approach: (1) an ensemble of teachers is trained on disjoint subsets of the sensitive data, (2) a student model is trained on public data labeled using the ensemble.

(2) [Knowledge distillation from few samples][22]



➔ 결국 훈련 데이터가 있긴 있다

➔ 도대체 훈련 데이터 업이 뭘 어떻게 한다는걸까? 상상해보십시오

3 Proposed Approach: KegNet (Knowledge Extraction with Generative Networks)

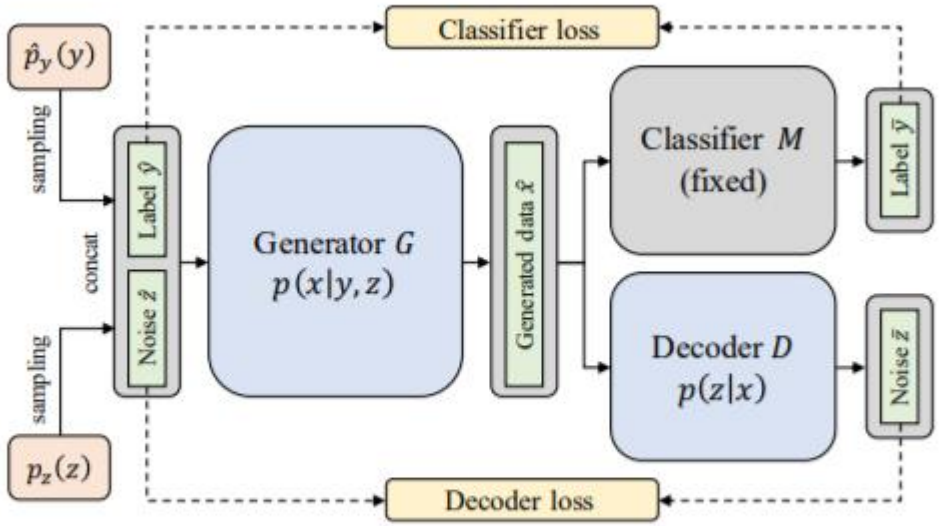


Figure 1: An overall structure of KEGNET. The generator creates artificial data and feed them into the classifier and decoder. The fixed classifier produces the label distribution of each data point, and the decoder finds its hidden representation as a low-dimensional vector.

$$l(\mathcal{B}) = \sum_{(\hat{y},\hat{z})} \left(l_{\text{cls}}(\hat{y},\hat{z}) + \alpha l_{\text{dec}}(\hat{y},\hat{z}) \right) + \beta l_{\text{div}}(\mathcal{B})$$

간단하게: 총 3개의 네트워크
Generator / Decoder / Classifier M (T)

어떻게 M의 지식이 담긴 X hat을 생성할 것인가?

3 Proposed Approach: Objective and Notation

$$l(\mathcal{B}) = \sum_{(\hat{y},\hat{z})} \left(l_{\text{cls}}(\hat{y},\hat{z}) + \alpha l_{\text{dec}}(\hat{y},\hat{z}) \right) + \beta l_{\text{div}}(\mathcal{B})$$

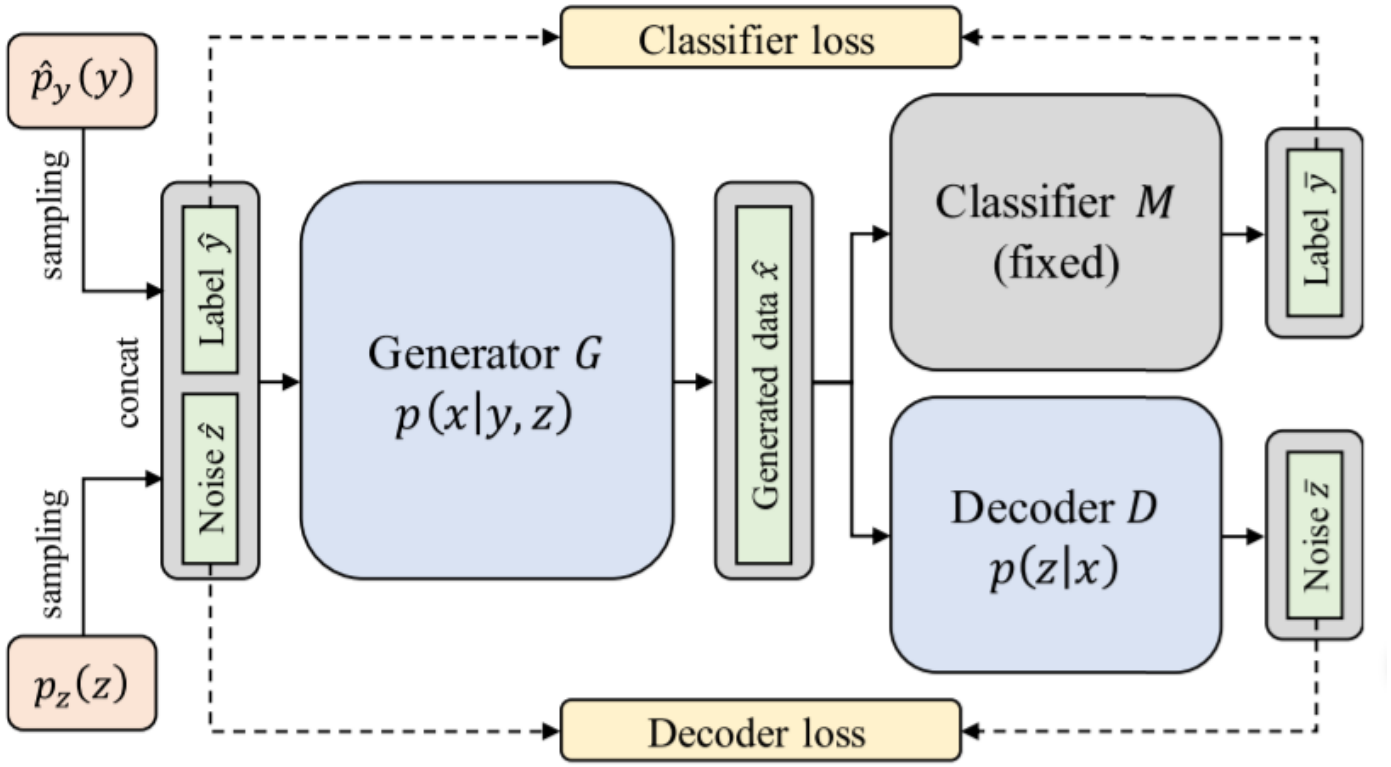
목표:
학습되고 고정된 모델 M을 입력으로 받아서 M의 Knowledge를 distill하기에 좋은 dataset D를 생성하는 것

Every yi is a (one-hot or soft) label vector
Every xi has a high conditional probability $p\,(xi|\ yi\ )$
Set $\mathcal{D}$ = {$(xi\ ,\ yi)|\ i$ = 1, 2, … } of artificial data

$$\mathcal{D} = \left\{ \text{argmax}_{\hat{x}}\, p(\hat{x}|\hat{y},\hat{z}) \;\middle|\; \hat{y} \sim \hat{p}_y(y) \text{ and } \hat{z} \sim p_z(z) \right\}$$

$\hat{p}_y$ and p_z are proposed distributions for $\hat{y}$ and z

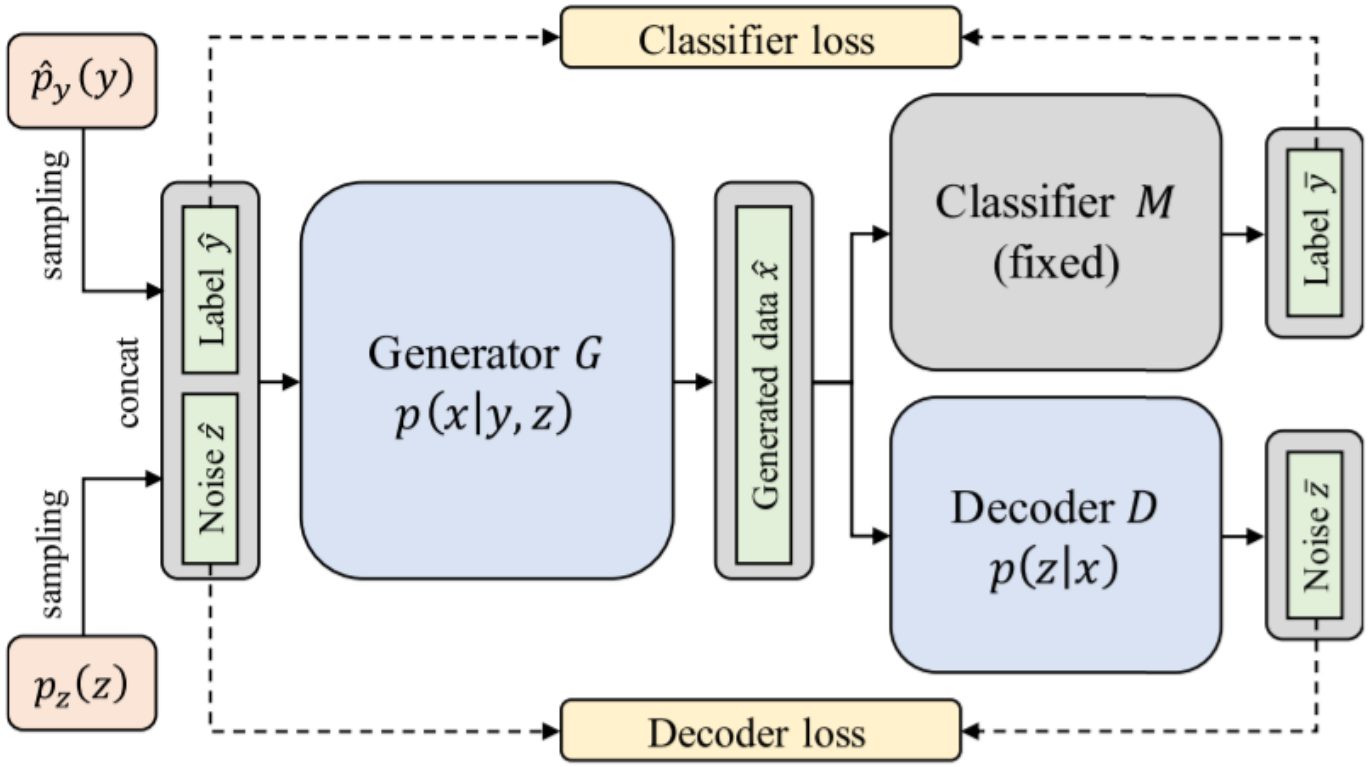
3 Proposed Approach: Objective and Notation



$$\mathcal{D} = \left\{ \text{argmax}_{\hat{x}}\, p(\hat{x}|\hat{y},\hat{z}) \;\middle|\; \hat{y} \sim \hat{p}_y(y) \text{ and } \hat{z} \sim p_z(z) \right\}$$

$\hat{p}_y$ and p_z are proposed distributions for $\hat{y}$ and z

3 Proposed Approach: Objective and Notation



따라서
Data Set D를 만드는 목적함수가
KegNet의 목적함수이며

y hat와 y (predicted)의 loss
Z hat와 z (predicted)의 loss를 통해
최적화할 수 있다.

Sampling $\hat{y}$ and $\hat{z}$ from $\hat{p}_y$ and p_z, resp.

Generating $\hat{x}$ from sampled $\hat{y}$ and $\hat{z}$

Reconstructing $\hat{y}$ from $\hat{x}$ (max. $p(\hat{y}|\hat{x})$)

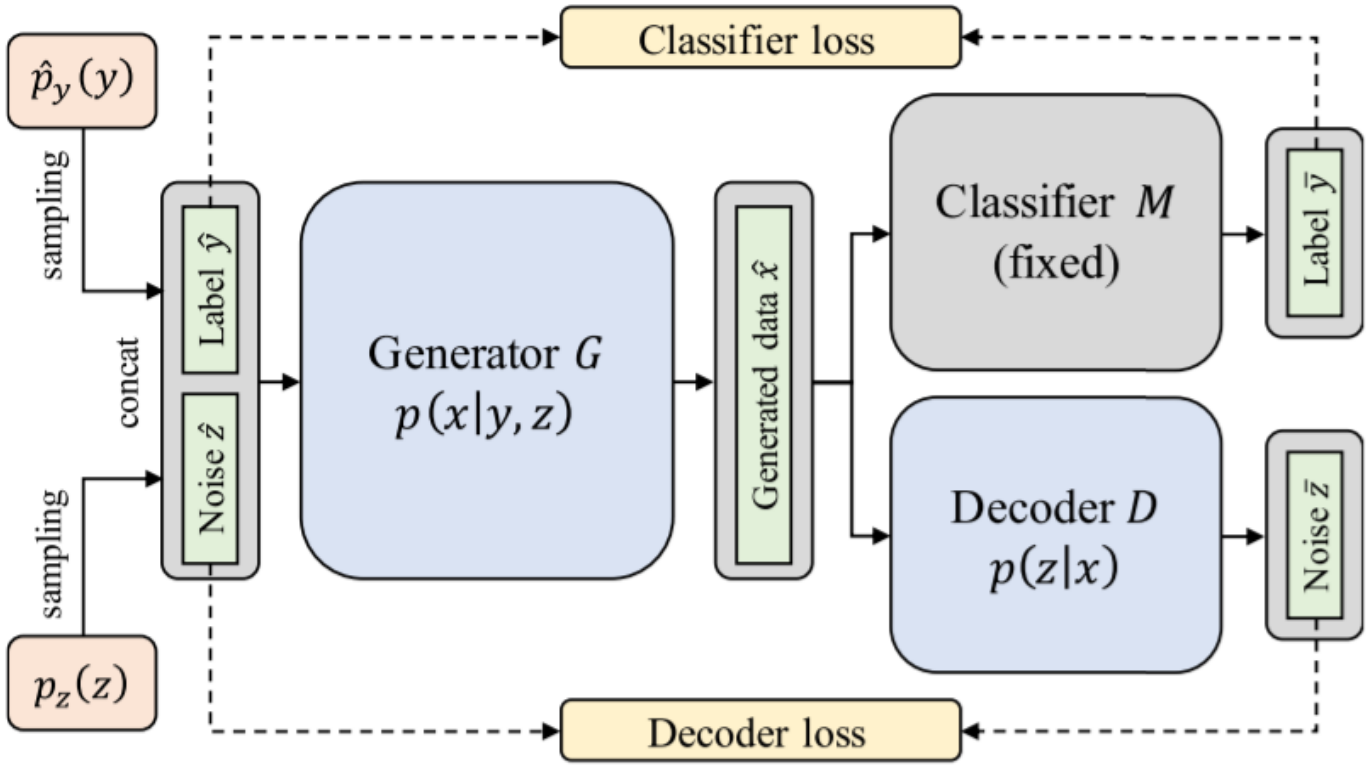
Reconstructing $\hat{z}$ from $\hat{x}$ (max. $p(\hat{z}|\hat{x})$)

$$\text{argmax}_{\hat{x}}\, p(\hat{x}|\hat{y},\hat{z}) \approx \text{argmax}_{\hat{x}}(\log p(\hat{y}|\hat{x}) + \log p(\hat{z}|\hat{x}))$$

근사하는 방법?

$\hat{p}_y$ and p_z are proposed distributions for $\hat{y}$ and z

3 Proposed Approach: Objective and Notation



Sample $\hat{y}$ and $\hat{z}$ from simple distributions

Convert variables by deep neural networks

Generator (to learn): $(\hat{y}, \hat{z}) \rightarrow \hat{x}$

Decoder (to learn): $\hat{x} \rightarrow \bar{z}$

Classifier (given and fixed): $\hat{x} \rightarrow \bar{y}$

Classifier loss: the distance $\hat{y} \leftrightarrow \bar{y}$

Decoder loss: the distance $\hat{z} \leftrightarrow \bar{z}$

Sampling $\hat{y}$ and $\hat{z}$ from $\hat{p}_y$ and p_z, resp.

Generating $\hat{x}$ from sampled $\hat{y}$ and $\hat{z}$

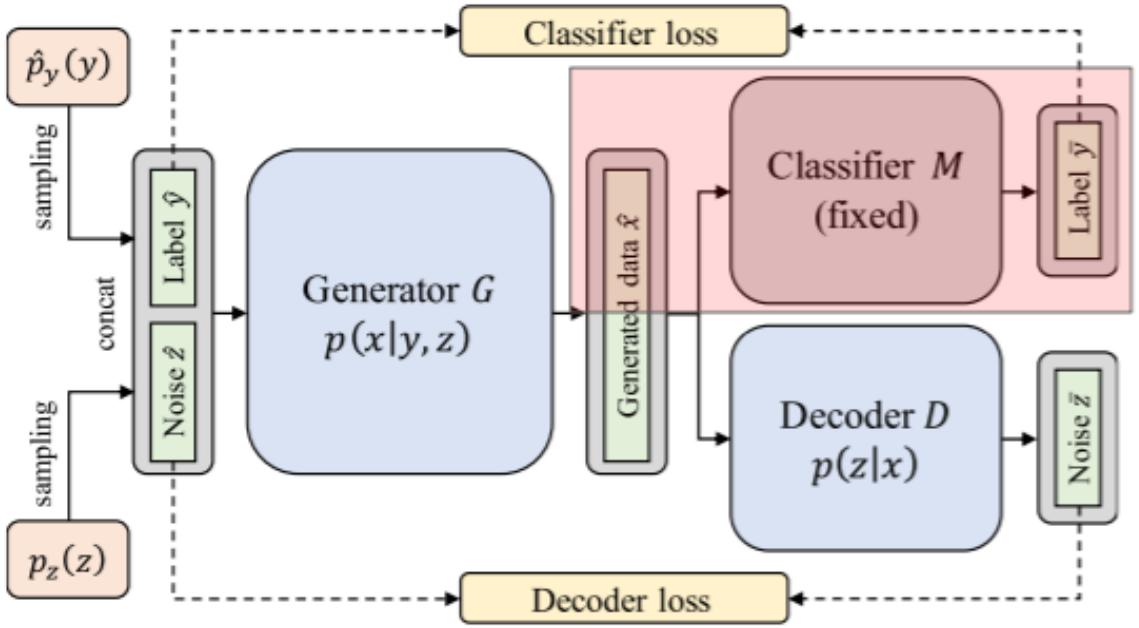
Reconstructing $\hat{y}$ from $\hat{x}$ (max. $p(\hat{y}|\hat{x})$)

Reconstructing $\hat{z}$ from $\hat{x}$ (max. $p(\hat{z}|\hat{x})$)

$$\text{argmax}_{\hat{x}}\, p(\hat{x}|\hat{y},\hat{z}) \approx \text{argmax}_{\hat{x}}(\log p(\hat{y}|\hat{x}) + \log p(\hat{z}|\hat{x}))$$

$\hat{p}_y$ and p_z are proposed distributions for $\hat{y}$ and z

3 Proposed Approach



Classifier loss: the distance $\hat{y} \leftrightarrow \bar{y}$

$$l_{\text{cls}}(\hat{y}, \hat{z}) = -\sum_{i \in \mathcal{S}} \hat{y}_i \log M(G(\hat{y}, \hat{z}))_i$$

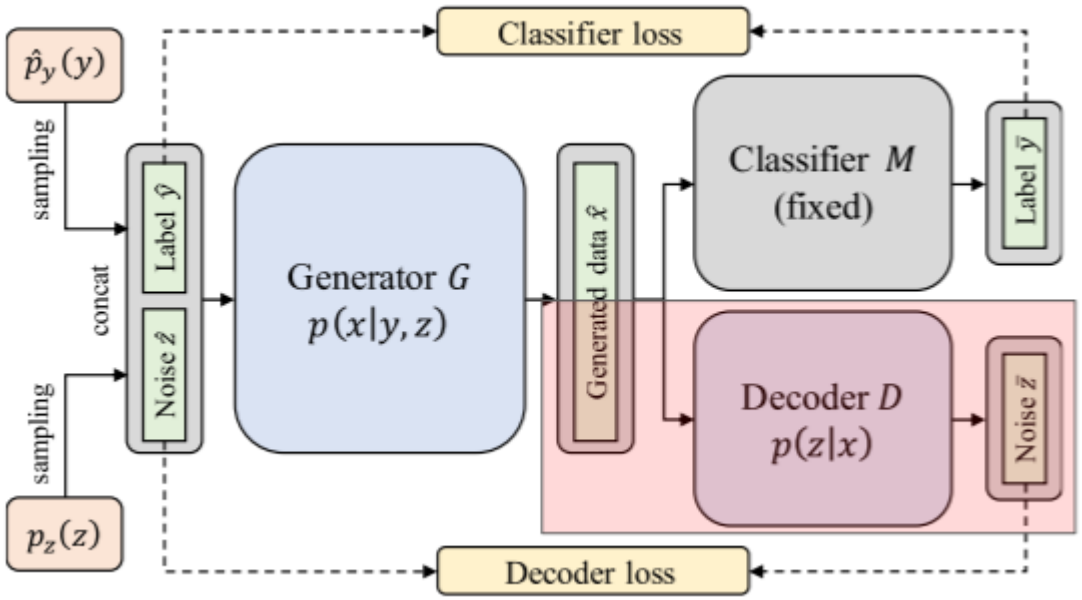
두 확률분포에 대한CE Loss로써
input label y'와 M(G(ˆy, zˆ)) 즉 y''의 LOSS

→ M이 잘 분류하는 매니폴드를 따르는 데이터 생성

→ 실제 X의 분포랑은 다를 수 있지만
최소한 M의 지식 EXTACT 가능

문제점: INPUT Y'에 대하여 똑같은 X' 만이 생성된다 -> LOSS dec

3 Proposed Approach



Decoder loss: the distance $\hat{z} \leftrightarrow \bar{z}$

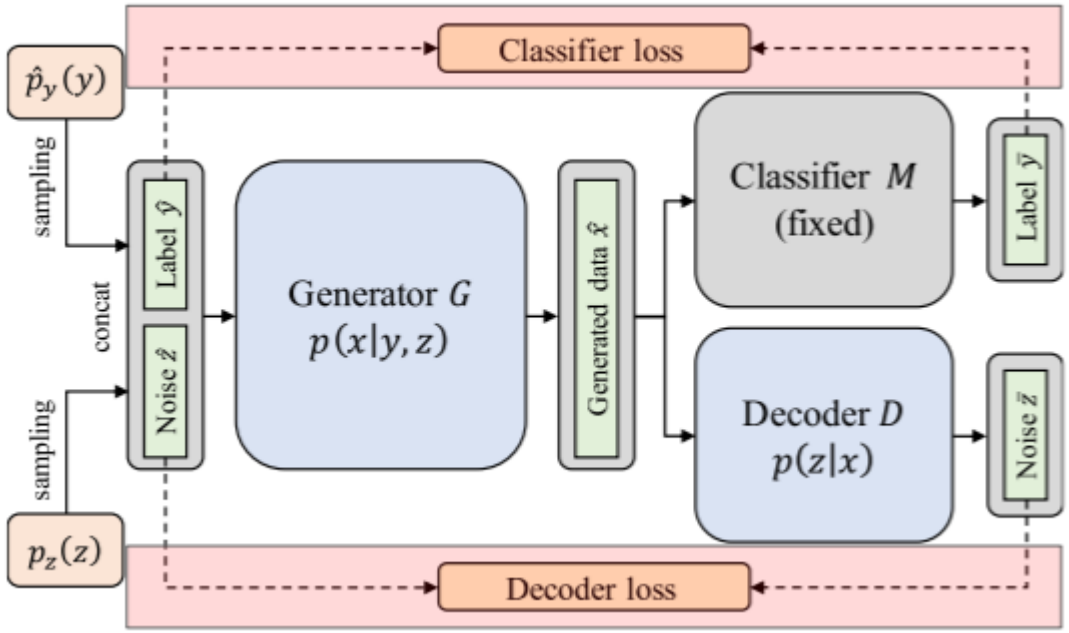
$$l_{\text{dec}}(\hat{y}, \hat{z}) = \|\hat{z} - D(G(\hat{y}, \hat{z}))\|_2^2.$$

G가 Z의 정보를 포함해서 생성하도록 함으로써 생성되는 데이터들이 동일한 Class에서도 다양성을 가지도록 하며

이때 디코더의 구조는 단순한 MLP 구조이다.

문제점: 그럼에도 동일한 Class에서의 데이터 다양성이 아직 부족하다.

3 Proposed Approach



온전하게 정보량을 많이 전달할 수 있는 데이터가 되기위해선 해당 데이터셋이 더 많은 latent space를 감당해야한다.

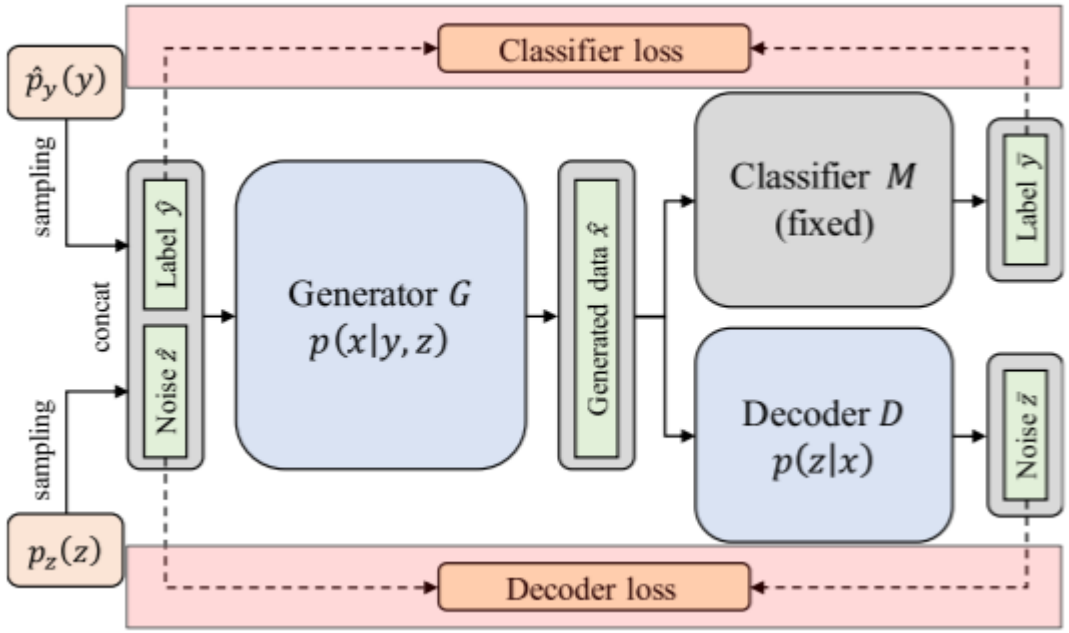
따라서, 해당 방법론은 그러한 다양성을 제어할 수 있는 Diversity Loss를 적용한다.

$$l_{\text{div}}(\mathcal{B}) = \exp\left(-\sum_{(\hat{z}_1,\hat{x}_1)}\sum_{(\hat{z}_2,\hat{x}_2)} \|\hat{z}_1 - \hat{z}_2\| \cdot d(\hat{x}_1,\hat{x}_2)\right)$$

따라서, 최종 loss는 다음과 같다.
(α ,β 는 하이퍼파라미터이다)

$$l(\mathcal{B}) = \sum_{(\hat{y},\hat{z})}\left(l_{\text{cls}}(\hat{y},\hat{z}) + \alpha l_{\text{dec}}(\hat{y},\hat{z})\right) + \beta l_{\text{div}}(\mathcal{B})$$

3 Proposed Approach



온전하게 정보량을 많이 전달할 수 있는 데이터가 되기위해선 해당 데이터셋이 더 많은 latent space를 감당해야한다.

따라서, 해당 방법론은 그러한 다양성을 제어할 수 있는 Diversity Loss를 적용한다.

$$l_{\text{div}}(\mathcal{B}) = \exp\left(-\sum_{(\hat{z}_1,\hat{x}_1)}\sum_{(\hat{z}_2,\hat{x}_2)} \|\hat{z}_1 - \hat{z}_2\| \cdot d(\hat{x}_1,\hat{x}_2)\right)$$

따라서, 최종 loss는 다음과 같다.
(α ,β 는 하이퍼파라미터이다)

$$l(\mathcal{B}) = \sum_{(\hat{y},\hat{z})}\left(l_{\text{cls}}(\hat{y},\hat{z}) + \alpha l_{\text{dec}}(\hat{y},\hat{z})\right) + \beta l_{\text{div}}(\mathcal{B})$$

<Distill Loss>

$$l_{\text{dis}}(\hat{y},\hat{z}) = \sum_{G\in\mathcal{G}} \text{CE}(M(G(\hat{y},\hat{z})), S(G(\hat{y},\hat{z}))),$$

4 Experiments: 실험개요

실험의 의도: 그럴듯한 데이터를 생성할 수 있는가
(=정보량이 더 많아서 KD를 더 잘할 수 있는 데이터인가?)

Experiments (1) : 임의의 데이터 vs 제안된 방법론을 통해 만들어진 데이터 비교

경량화된 네트워크 Student에 KD했을 때 더 높은 정확도를 보이는 데이터가
KD에 사용된 데이터의 품질(유사성, 정보전달 가능량)이 높다고 생각할 수 있다.

Experiments (2): 실제 생성된 데이터를 정성적으로 품질 확인

경량화 : Tucker Decomp(터커 분해)
분류모델: LeNet5, ResNet14
분류모델의 학습데이터셋: MNIST / SVHN / Fasion MNIST

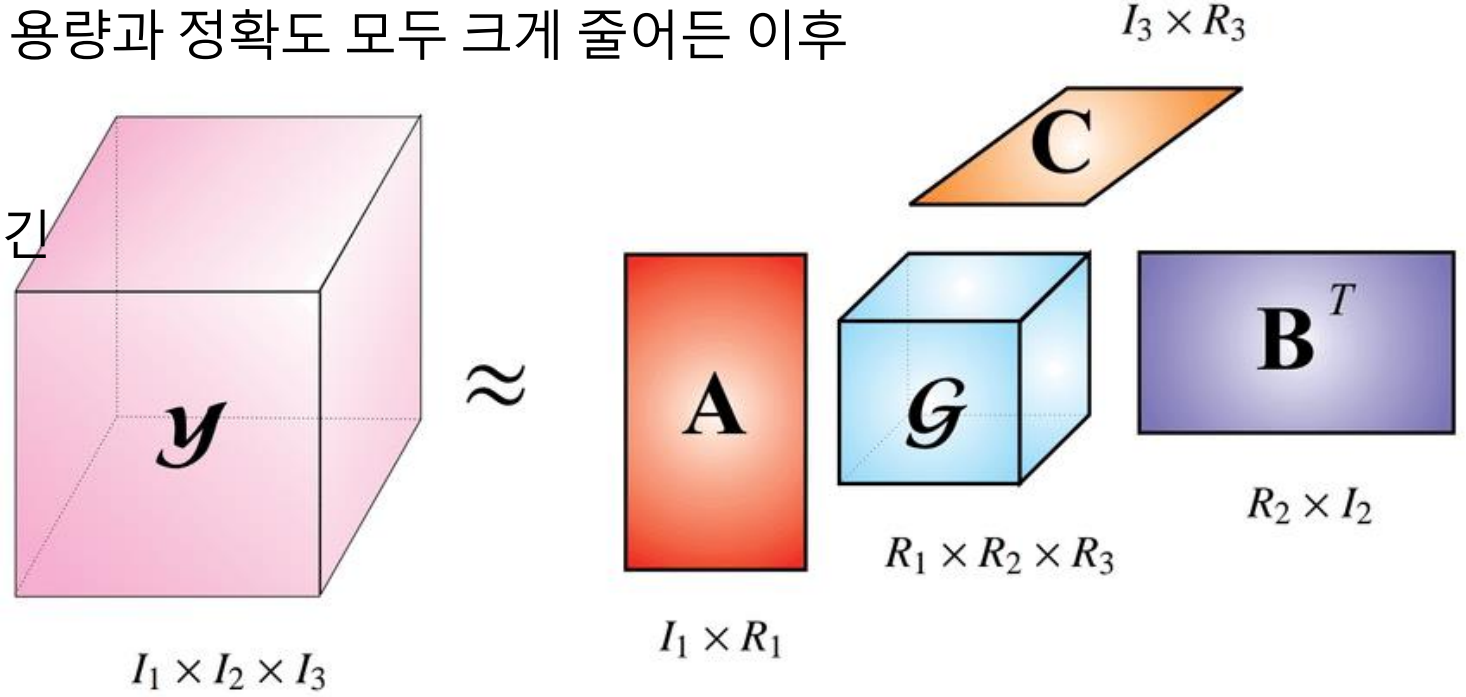
4 SIL Experiments: Tucker decomposition

터커 분해는 네트워크의 가중치(행렬,즉 Tensor)을 분해함으로써 압축할 수 있다.

기본적으로 하나의 Tensor를 하나의 작은 core Tenser와 matric들로 분해한다.
요인분석과 주성분분석의 확장판으로 볼 수 있으며 HOSVD 분해의 일반화형태로 쓰인다. (위키피디아)

데이터가 없는 상황에서도 가능하지만 , 압축 이후 Fine-Tune을 하지 않으면 모델의 성능이 크게 훼손된다.
따라서, Tucker Decomp로 압축하여 모델의 용량과 정확도 모두 크게 줄어든 이후

실험 데이터를 통해 KD를 진행하고 성능의
회복정도를 통해 KD에 사용된 데이터에 담긴
Knowledge의 양을 판단할 수 있다.



4 Experiments: 실험 결과 (1)

Experiments (1) : 임의의 데이터 vs 제안된 방법론을 통해 만들어진 데이터 비교

Dataset	Model	Approach	Student 1	Student 2	Student 3
MNIST	LeNet5	Original	98.90%	98.90%	98.90%
MNIST	LeNet5	Tucker (T)	85.18% (3.62×)	67.35% (4.10×)	50.01% (4.49×)
MNIST	LeNet5	T+Uniform	95.48 ± 0.11%	88.27 ± 0.07%	69.89 ± 0.28%
MNIST	LeNet5	T+Gaussian	95.45 ± 0.15%	87.70 ± 0.12%	71.76 ± 0.18%
MNIST	LeNet5	T+KEGNET	**96.32 ± 0.05%**	**90.89 ± 0.11%**	**89.94 ± 0.08%**
SVHN	ResNet14	Original	93.23%	93.23%	93.23%
SVHN	ResNet14	Tucker (T)	19.31% (1.44×)	11.02% (1.65×)	11.07% (3.36×)
SVHN	ResNet14	T+Uniform	33.08 ± 1.47%	63.08 ± 1.77%	23.83 ± 1.86%
SVHN	ResNet14	T+Gaussian	26.58 ± 1.61%	60.22 ± 4.17%	21.49 ± 2.96%
SVHN	ResNet14	T+KEGNET	**69.89 ± 1.24%**	**87.26 ± 0.46%**	**63.40 ± 1.80%**
Fashion	ResNet14	Original	92.50%	92.50%	92.50%
Fashion	ResNet14	Tucker (T)	65.09% (1.40×)	75.80% (1.58×)	46.55% (2.90×)
Fashion	ResNet14	T+Uniform	< 65.09%	< 75.80%	< 46.55%
Fashion	ResNet14	T+Gaussian	< 65.09%	< 75.80%	< 46.55%
Fashion	ResNet14	T+KEGNET	**85.23 ± 1.36%**	**87.80 ± 0.31%**	**79.95 ± 1.36%**

해당 실험에서

임의의 데이터

Uniform: 균등분포
P(x) ~ U(-1, 1)를 따르는 데이터
Normal: 정규분포
p(x) ~ N(0, 1)를 따르는 데이터

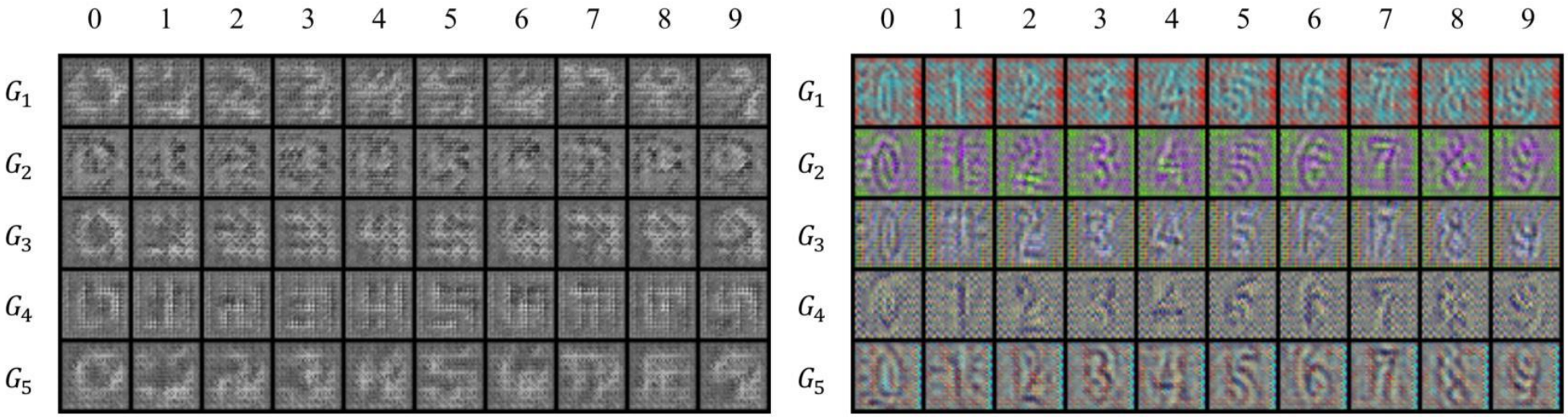
그리고 **KagNet**이 생성한 데이터

비교를 통해

KagNet이 더 정보량이 많은
데이터를 생성했음을 알 수 있다.

4 Experiments: 실험 결과 (2)

Experiments (2): 실제 생성된 데이터를 정성적으로 품질 확인

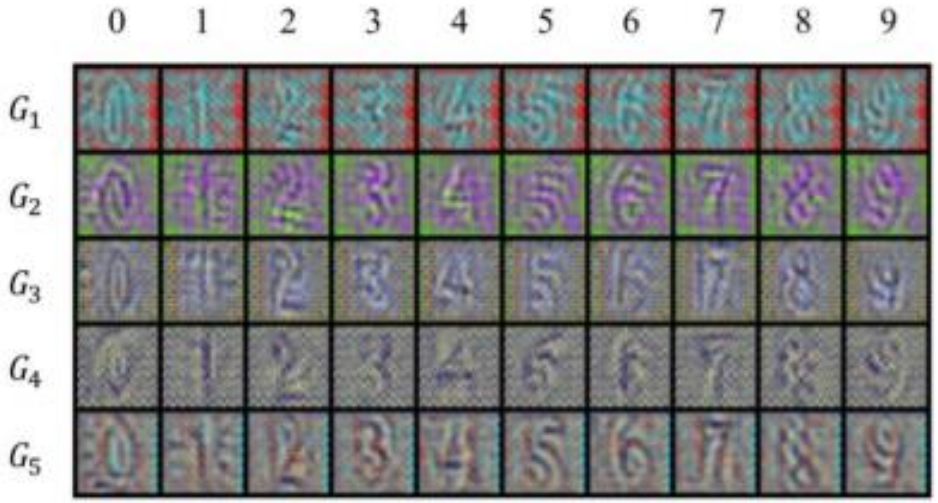


(a) MNIST ($z = \mathbf{0}$). (b) SVHN ($z = \mathbf{0}$).

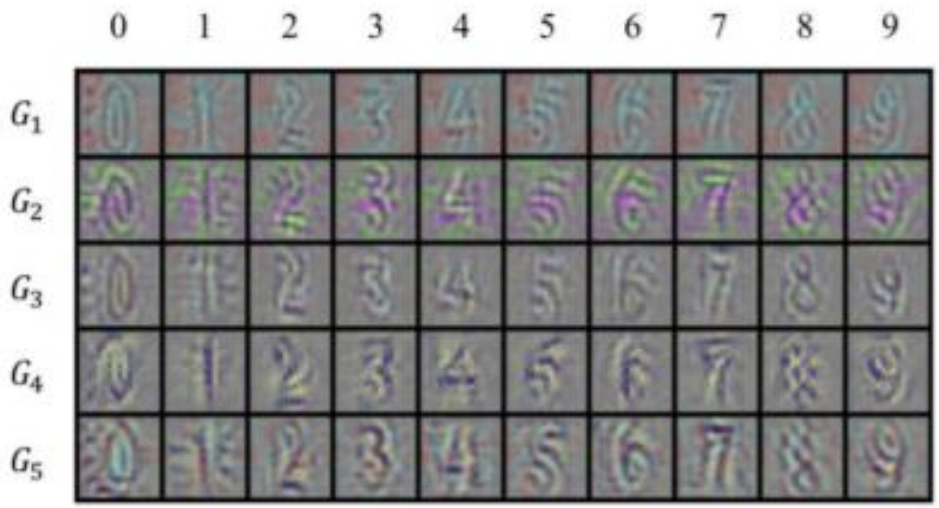
Y 라벨에 맞는 이미지가 생성되는 것을 보아
KegNet이 분포 모델이 학습하는데 사용한 데이터와 유사한 데이터를 잘 생성했음을 알 수 있다.

4 Experiments: 실험 결과 (2)

Experiments (2): 실제 생성된 데이터를 정성적으로 품질 확인



(b) SVHN ($z = \mathbf{0}$).



(c) SVHN (averaged by z).

1. SVHN이 MNIST보다 선명하다.
2. z가 이미지에 변동성(다양성)을 준다.

5 CONCLUSION

"입력된 분류 모델을 기반으로 새로운 생성망(Generater)을 학습함으로써

분류 모델이 기조에 학습되었던 데이터의 분포를 추론하여

비슷한 분포를 가지는 그럴듯한 유사 데이터를 생성하여, 지식 증류를 가능하게 한다."
- 실제 해당 연구를 진행한 연구실 (https://snudmlab.blogspot.com/)

Classifier network which is given and fixed

Generator network for generating artificial data

Decoder network for capturing latent variables

KEGNET (Knowledge Extraction with Generative Networks), a novel architecture
That extracts the knowledge of a trained neural network without any observable data.

KEGNET learns the conditional distribution of data points by training the generator and decoder networks, and estimates the manifold of missing data as a set of artificial data points.

감사합니다