Semi-supervised Learning with Deep Generative Models

Kingma, Diederik P., et al. "Semi-supervised learning with deep generative models." *Advances in Neural Information Processing Systems*. 2014.

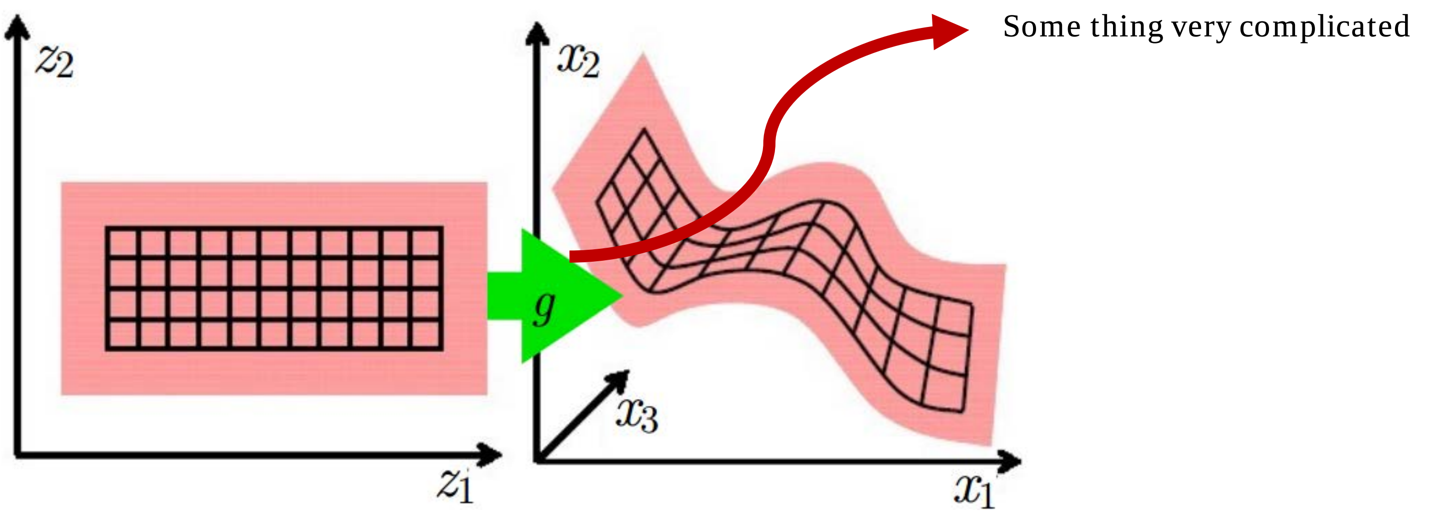
2015.11.13 산업공학특론 발표

Latent variable model

- Latent variables can extract the true explanatory factors of the (observed) original variables : generative model
- Latent variable space tends to a more simple space.

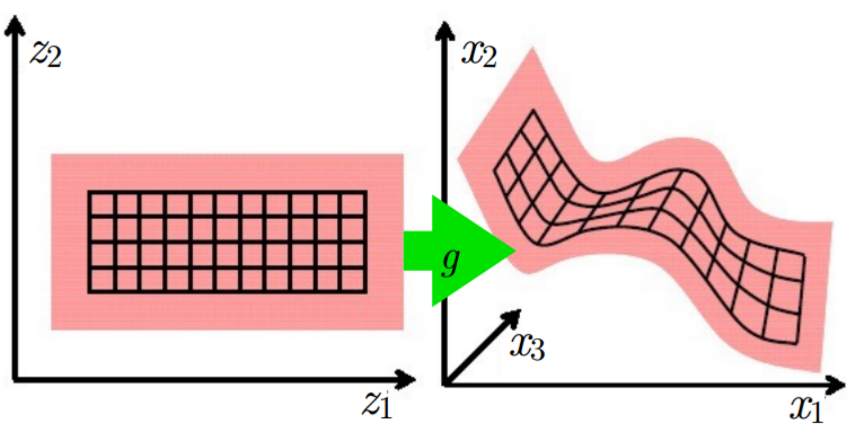
$$p(x) = \int p(x, z)\, dz \quad \text{where} \quad p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x} \mid \boldsymbol{z})p(\boldsymbol{z})$$

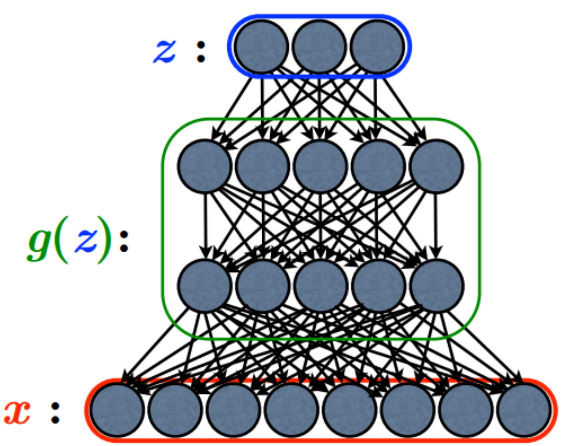
$$p(\boldsymbol{z}) = \text{something simple} \qquad p(\boldsymbol{x} \mid \boldsymbol{z}) = g(\boldsymbol{z})$$



Latent variable model

- Use neural networks as the (generative) transformation g from the latent space to the original feature space.
- For both training and inference, latent variable z must be inferred.
- This is a generative model : inferring the latent variable is hard
- The posterior $p_\theta(z|x)$ is intractable





Variational inference

- How to infer the latent variable z
- Use $q_\varphi(z|x)$ as an alternative to $P_\theta(z|x)$
- Maximize the lower bound of the likelihood $p_\theta(x) \geq L(\theta, \varphi, x)$

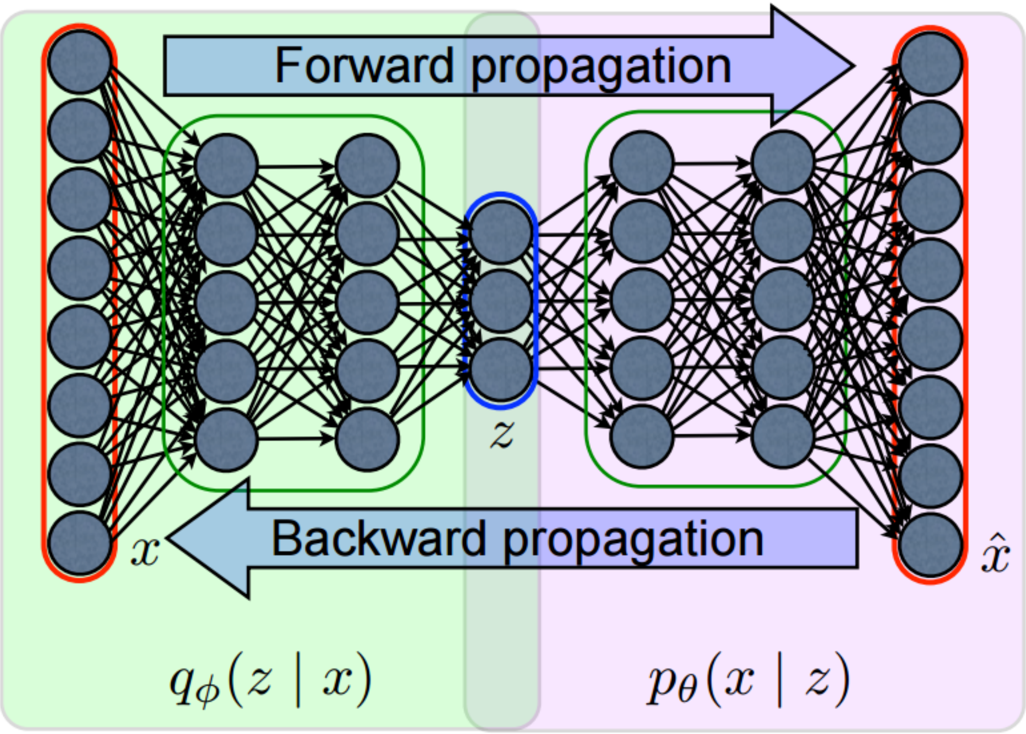
$$\begin{aligned}
\mathcal{L}(\theta, \phi, x) &= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x, z) - \log q_\phi(z \mid x)\right] \\
&= \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z) + \log p_\theta(z) - \log q_\phi(z \mid x)\right] \\
&= -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right)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

***regularization* term** ***reconstruction* term**

- Train this by using backpropgation

Variational Autoencoder(VAE)

Objective function: $\mathcal{L}(\theta, \phi, x) = -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z \mid x)}\left[\log p_\theta(x \mid z)\right]$



Variational Autoencoder(VAE)

- Since we assume that $P_\theta(z|x)$ is a complex non-closed function, Monte Carlo methods should be used.

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right]$$

$$\widetilde{\mathcal{L}}^A(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \frac{1}{L}\sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})$$

- $z^{(i,l)} \sim q_\varphi(z|x)$

- We can’t do proper backpropagation
  - The parameter φ is involved in the sampling procedure that can’t be differentiated.

Reparameterization trick

- Parametrize the distribution $q_\varphi(z|x)$ by deep neural networks and do Monte Carlo.
- $q_\varphi(z|x) = N(z|\mu_z(x), \sigma_z(x))$
- z = $\mu_z(x) + \sigma_z(x)\varepsilon_z$ where $\varepsilon_z \sim N(0,1)$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})}\left[-\log q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) + \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\right]$$

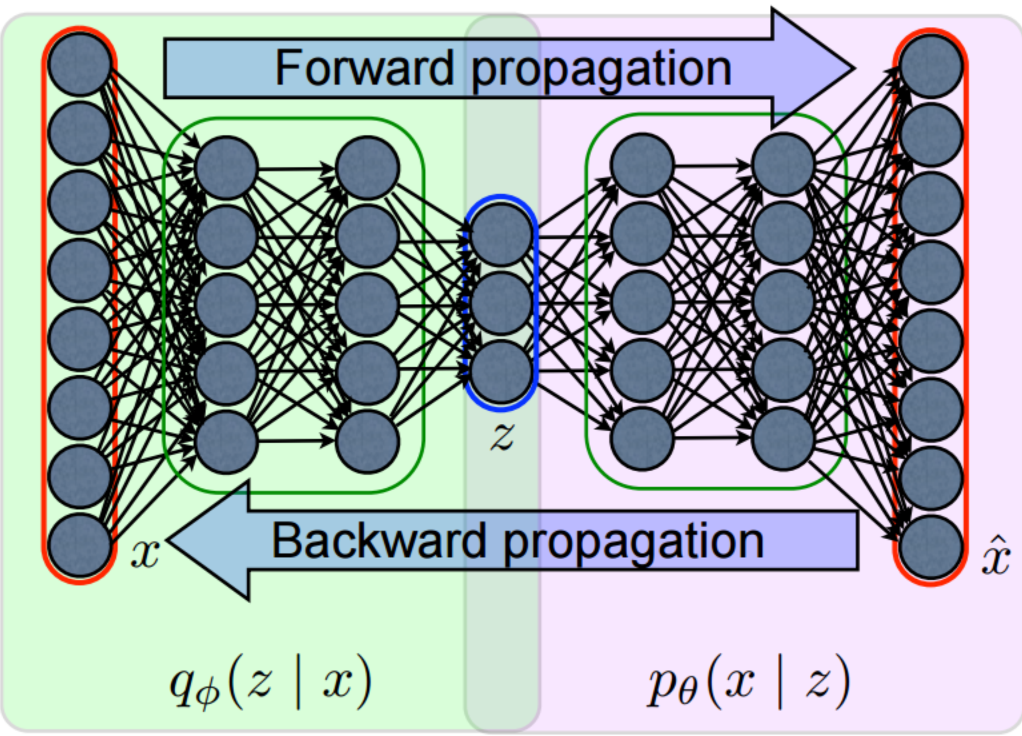
$$\widetilde{\mathcal{L}}^A(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)}) = \frac{1}{L}\sum_{l=1}^{L} \log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}, \mathbf{z}^{(i,l)}) - \log q_{\boldsymbol{\phi}}(\mathbf{z}^{(i,l)}|\mathbf{x}^{(i)})$$

$$\text{where} \quad \mathbf{z}^{(i,l)} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(i,l)}, \mathbf{x}^{(i)}) \quad \text{and} \quad \boldsymbol{\epsilon}^{(l)} \sim p(\boldsymbol{\epsilon})$$

- Sampling doesn’t involve the parameter.

Variational Autoencoder

Objective function: $\mathcal{L}(\theta, \phi, x) = -D_{\mathrm{KL}}\left(q_\phi(z \mid x) \| p_\theta(z)\right) + \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]$



Semi-supervised learning with deep generative models

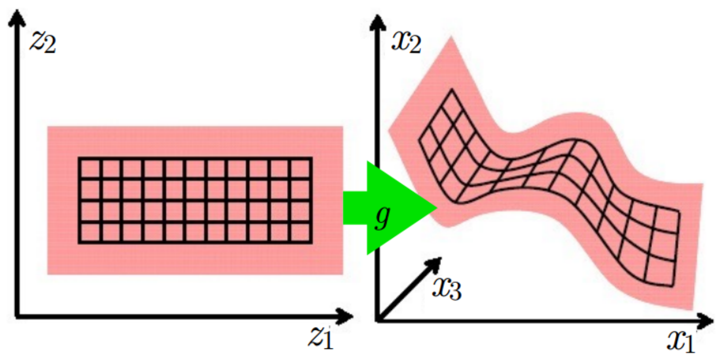
- Semi-supervised learning : few labeled data, abundant labeled data
- Develop a generative model that combines probabilistic modelling and deep neural networks
  - Generative Latent-feature model(M1) : used to extract features
  - Generative semi-supervised model(M2) : used for classification
- Implement variational autoencoders in each model

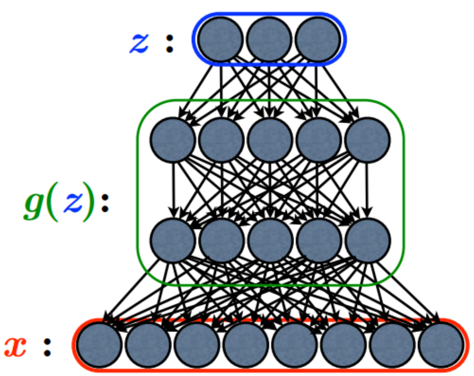
Generative Latent-feature model(M1)

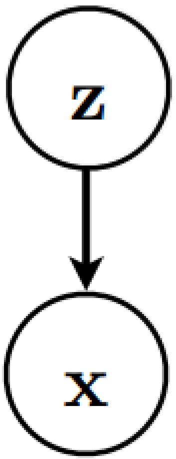
- Unsupervised feature learning using a generative model

  $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}); \qquad p_\theta(\mathbf{x}|\mathbf{z}) = f(\mathbf{x}; \mathbf{z}, \boldsymbol{\theta})$

- f is a non-linear transformation of the latent variable z with parameters θ : use deep neural networks
- Exactly equal to the variational autoencoder







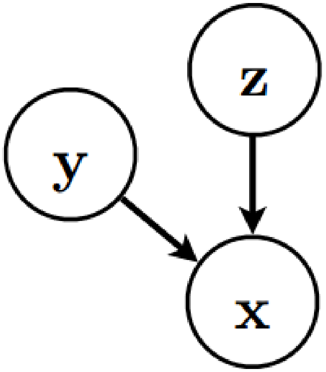
Generative semi-supervised model(M2)

- Semi-supervised feature learning using a generative model

$$p(y) = \mathrm{Cat}(y|\boldsymbol{\pi}); \qquad p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I});$$

$$p_\theta(\mathbf{x}|y, \mathbf{z}) = f(\mathbf{x}; y, \mathbf{z}, \boldsymbol{\theta}),$$

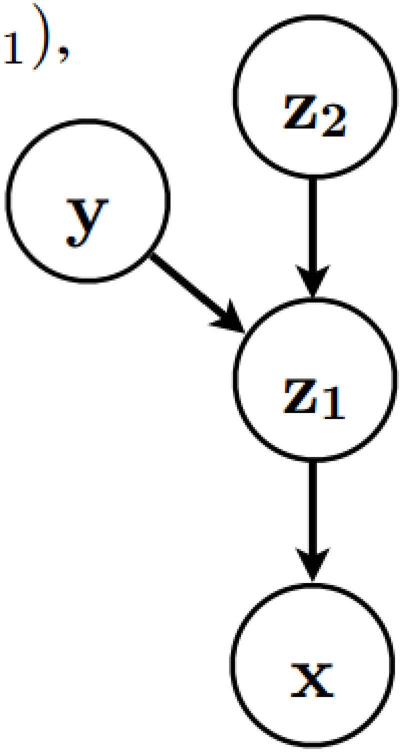
- Variational autoencoder with an additional information on labels
  - If the label is known : plug it in
  - If the label is unknown : an additional latent variable



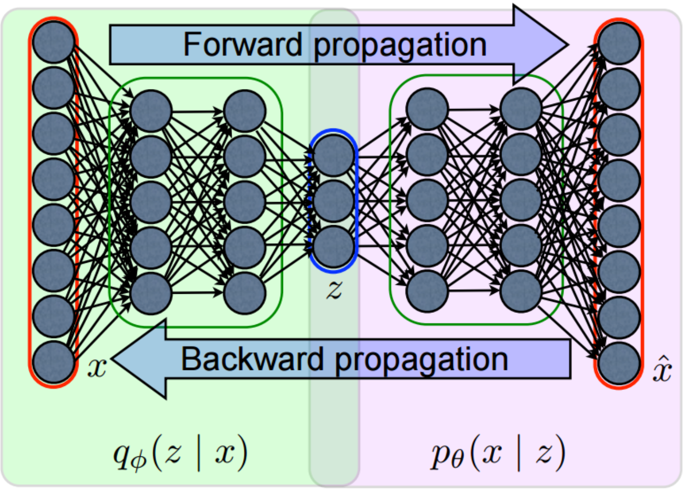
Stack of models M1, M2

- Train generative semi-supervised model(M2) on unsupervised features $z1$ from latent feature model (M1)

$$p_\theta(\mathbf{x}, y, \mathbf{z}_1, \mathbf{z}_2) = p(y)p(\mathbf{z}_2)p_\theta(\mathbf{z}_1|y, \mathbf{z}_2)p_\theta(\mathbf{x}|\mathbf{z}_1),$$



VAE on each model



M1: $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(\mathbf{x}), \text{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x}))),$

M2: $q_\phi(\mathbf{z}|y, \mathbf{x}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(y, \mathbf{x}), \text{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x}))); \quad q_\phi(y|\mathbf{x}) = \text{Cat}(y|\boldsymbol{\pi}_\phi(\mathbf{x})),$

- Lowerbound objective for latent feature model(M1)

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right] - KL[q_\phi(\mathbf{z}|\mathbf{x}) \| p_\theta(\mathbf{z})] = -\mathcal{J}(\mathbf{x}),$$

- Lowerbound objective for generative semi-supervised model(M2)
  - Labeled

$$\log p_\theta(\mathbf{x}, y) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},y)}\left[\log p_\theta(\mathbf{x}|y,\mathbf{z}) + \log p_\theta(y) + \log p(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x},y)\right] = -\mathcal{L}(\mathbf{x}, y),$$

  - Unlabeled

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q_\phi(y,\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|y,\mathbf{z}) + \log p_\theta(y) + \log p(\mathbf{z}) - \log q_\phi(y,\mathbf{z}|\mathbf{x})\right]$$
$$= \sum_y q_\phi(y|\mathbf{x})(-\mathcal{L}(\mathbf{x}, y)) + \mathcal{H}(q_\phi(y|\mathbf{x})) = -\mathcal{U}(\mathbf{x}).$$

Assume independence between y and z

  - Combined

$$\mathcal{J} = \sum_{(\mathbf{x},y)\sim\widetilde{p}_l} \mathcal{L}(\mathbf{x}, y) + \sum_{\mathbf{x}\sim\widetilde{p}_u} \mathcal{U}(\mathbf{x})$$

M1, M2 training

Algorithm 1 Learning in model M1

---

while generativeTraining() **do**
  $\mathcal{D} \leftarrow$ getRandomMiniBatch()
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|\mathbf{x}_i) \quad \forall \mathbf{x}_i \in \mathcal{D}$
  $\mathcal{J} \leftarrow \sum_n \mathcal{J}(\mathbf{x}_i)$
  $(\mathbf{g}_\theta, \mathbf{g}_\phi) \leftarrow (\frac{\partial \mathcal{J}}{\partial \theta}, \frac{\partial \mathcal{J}}{\partial \phi})$
  $(\boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow (\boldsymbol{\theta}, \boldsymbol{\phi}) + \boldsymbol{\Gamma}(\mathbf{g}_\theta, \mathbf{g}_\phi)$
end while
while discriminativeTraining() **do**
  $\mathcal{D} \leftarrow$ getLabeledRandomMiniBatch()
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|\mathbf{x}_i) \quad \forall \{\mathbf{x}_i, y_i\} \in \mathcal{D}$
  trainClassifier($\{\mathbf{z}_i, y_i\}$)
end while

Algorithm 2 Learning in model M2

---

while training() **do**
  $\mathcal{D} \leftarrow$ getRandomMiniBatch()
  $y_i \sim q_\phi(y_i|\mathbf{x}_i) \quad \forall \{\mathbf{x}_i, y_i\} \notin \mathcal{O}$
  $\mathbf{z}_i \sim q_\phi(\mathbf{z}_i|y_i, \mathbf{x}_i)$
  $\mathcal{J}^\alpha \leftarrow$ eq. (9)
  $(\mathbf{g}_\theta, \mathbf{g}_\phi) \leftarrow (\frac{\partial \mathcal{L}^\alpha}{\partial \theta}, \frac{\partial \mathcal{L}^\alpha}{\partial \phi})$
  $(\boldsymbol{\theta}, \boldsymbol{\phi}) \leftarrow (\boldsymbol{\theta}, \boldsymbol{\phi}) + \boldsymbol{\Gamma}(\mathbf{g}_\theta, \mathbf{g}_\phi)$
end while

Experimental results

Table 1: Benchmark results of semi-supervised classification on MNIST with few labels.

N	NN	CNN	TSVM	CAE	MTC	AtlasRBF	M1+TSVM	M2	M1+M2
100	25.81	22.98	16.81	13.47	12.03	8.10 ($\pm$ 0.95)	11.82 ($\pm$ 0.25)	11.97 ($\pm$ 1.71)	**3.33** ($\pm$ 0.14)
600	11.44	7.68	6.16	6.3	5.13	–	5.72 ($\pm$ 0.049)	4.94 ($\pm$ 0.13)	**2.59** ($\pm$ 0.05)
1000	10.7	6.45	5.38	4.77	3.64	3.68 ($\pm$ 0.12)	4.24 ($\pm$ 0.07)	3.60 ($\pm$ 0.56)	**2.40** ($\pm$ 0.02)
3000	6.04	3.35	3.45	3.22	2.57	–	3.49 ($\pm$ 0.04)	3.92 ($\pm$ 0.63)	**2.18** ($\pm$ 0.04)

- CNN : convolutional neural network
- TSVM : transductive SVM
- CAE : contrastive autoencoder
- MTC : Manifold tangent classifier (CAE based manifold learning method)
- AtlasRBF (graph-based semi-supervised learning method)

- TSVM with M1 features are better than TSVM with original features
- M1+M2 shows the best performance.

- M1 : MLP with two hidden layer with 600 hidden units, latent variable 50 dimensions.
- M2 : MLP with one hidden layer with 500 hidden units, latent variable 50 dimensions.

Conditional Generation

(a) Handwriting styles for MNIST obtained by fixing the class label and varying the 2D latent variable $\mathbf{z}$

Conditional Generation

(b) MNIST analogies

(c) SVHN analogies

Experimental Results on other datasets

- NORB dataset : images of 50 toys belonging to 5 generic categories
- SVHN datset : street view house numbers dataset

Table 2: Semi-supervised classification on the SVHN dataset with 1000 labels.

KNN	TSVM	M1+KNN	M1+TSVM	M1+M2
77.93 (± 0.08)	66.55 (± 0.10)	65.63 (± 0.15)	54.33 (± 0.11)	**36.02** (± 0.10)

Table 3: Semi-supervised classification on the NORB dataset with 1000 labels.

KNN	TSVM	M1+KNN	M1+TSVM
78.71 (± 0.02)	26.00 (± 0.06)	65.39 (± 0.09)	**18.79** (± 0.05)

On anomaly detection

- If labels on anomalies can be obtained, M1+M2 model can be used.
- Conditional generation with fixed labels can reveal types of variations in each anomaly label.
- Conditional generation with fixed latent variables can reveal the characteristics and structures of a given latent space among normal and different types of anomalies.

- The paper used the same number of instances for each class, so for unbalanced data, other treatments would be required.

Referenences

- Kingma, Diederik P., et al. "Semi-supervised learning with deep generative models." *Advances in Neural Information Processing Systems*. 2014.
- Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* (2013).