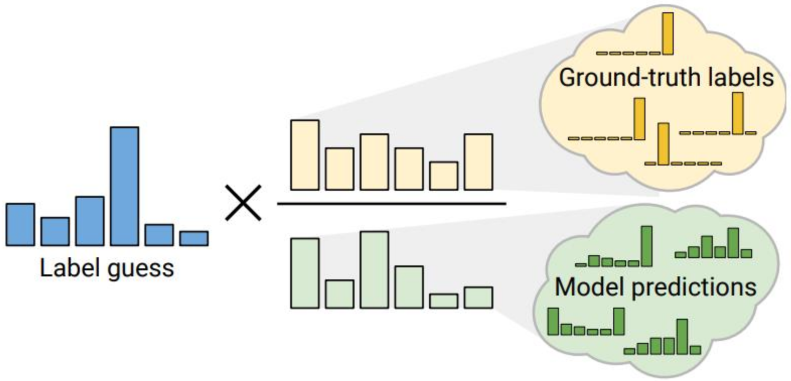
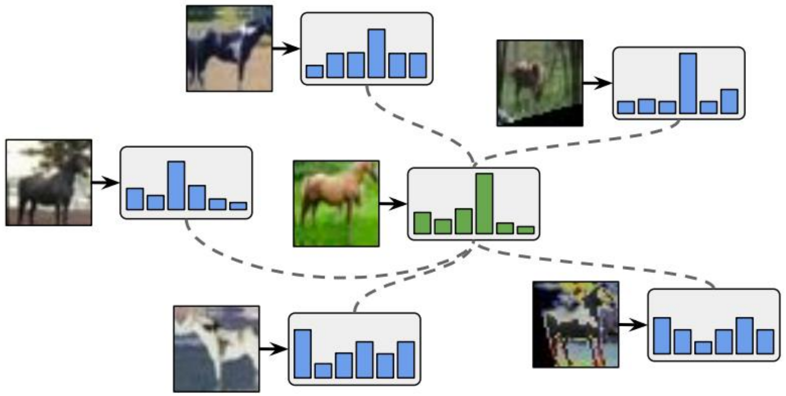


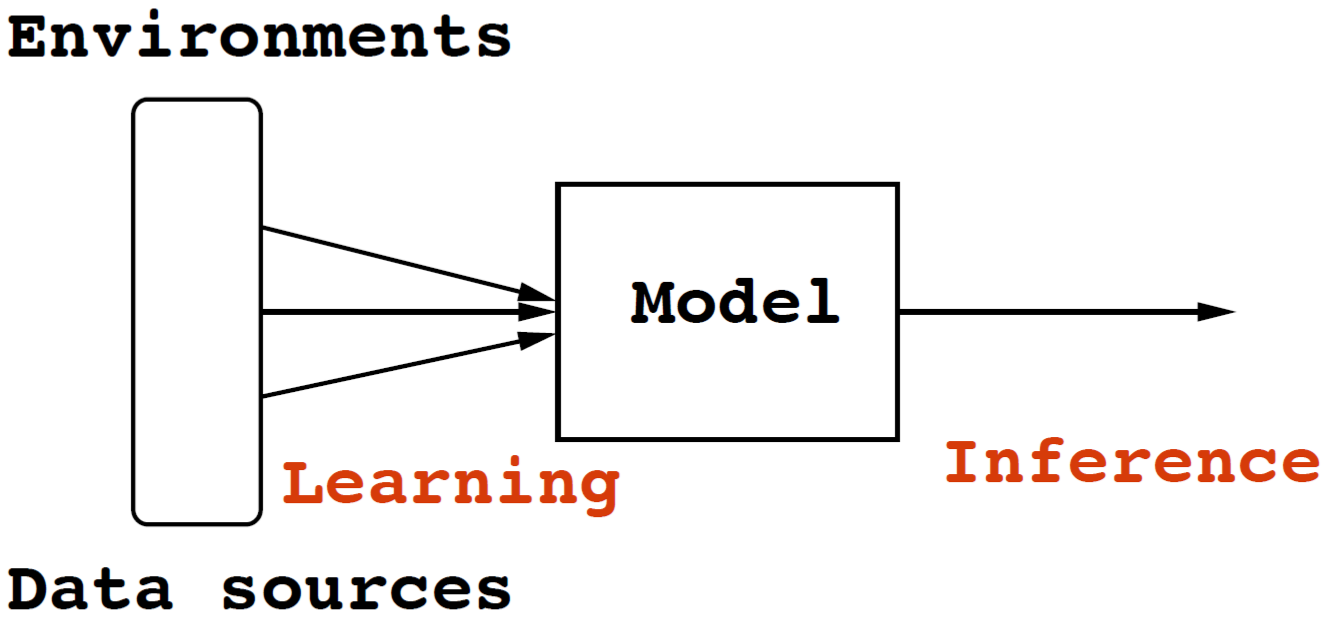


Semi-Supervised Learning: Generative Models

Pilsung Kang
School of Industrial Management Engineering
Korea University

Discriminative vs. Generative

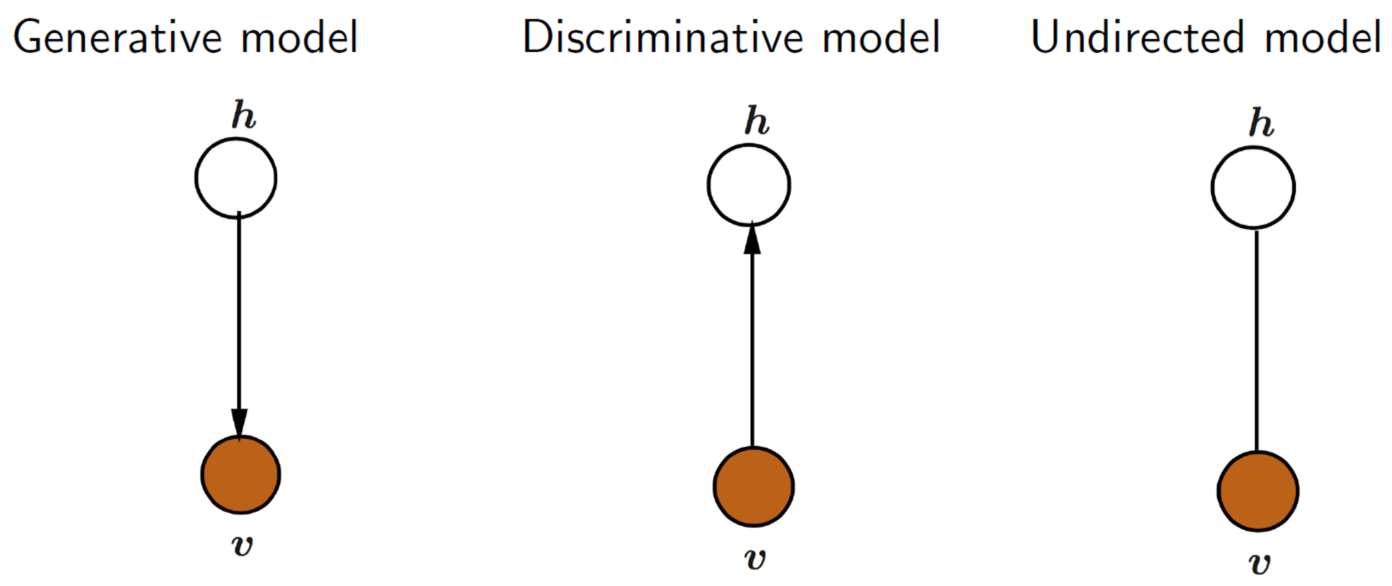
Choi (2015)



Machine Learning: A scientific discipline that is concerned with the design and development of algorithms that allow computers to learn from empirical data (sensor data or database) and to make predictions.

Discriminative vs. Generative

Choi (2015)



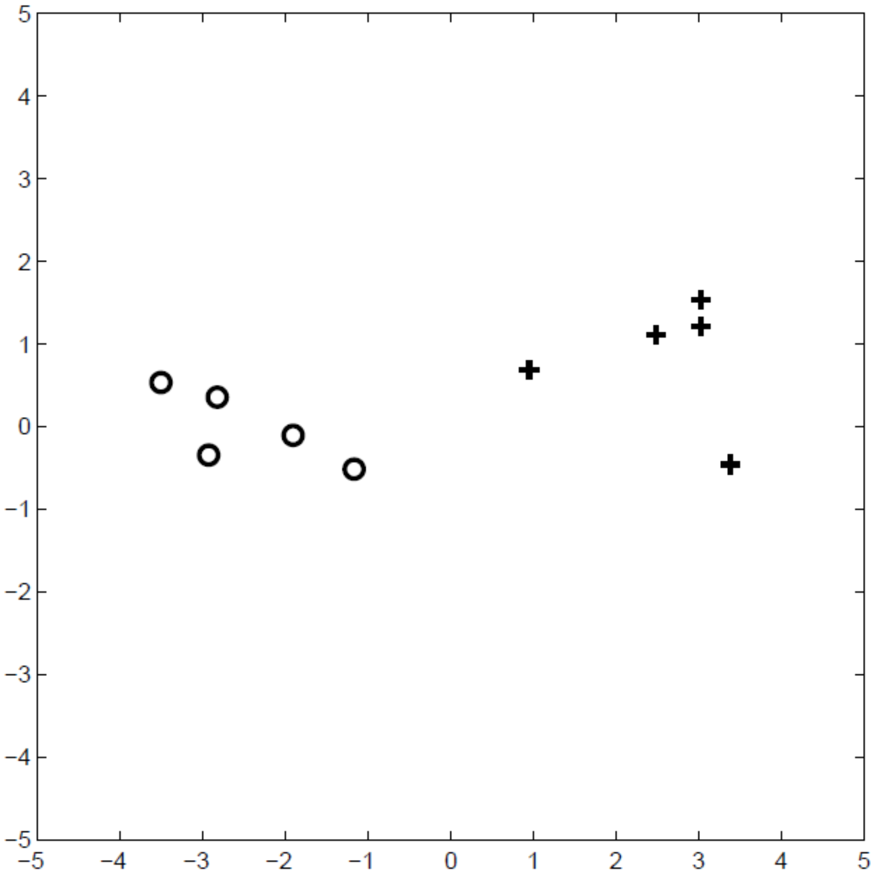
- Generative model: Joint distribution $p(\boldsymbol{v}, \boldsymbol{h}) = p(\boldsymbol{v}|\boldsymbol{h})p(\boldsymbol{h})$
- Discriminative model: Directly model $p(\boldsymbol{h}|\boldsymbol{v})$
- Undirected model: Energy-based model

$$p(\boldsymbol{v}, \boldsymbol{h}) = \frac{\exp\left\{-E\left(\boldsymbol{v}, \boldsymbol{h}\right)\right\}}{\sum_{\boldsymbol{v}', \boldsymbol{h}'} \exp\left\{-E\left(\boldsymbol{v}', \boldsymbol{h}'\right)\right\}}$$

Generative Models

Zhu (2007)

- Example
  - ✓ Assuming each class has a Gaussian distribution, what is the decision boundary?



Model parameters :

$$\theta = \{w_1, w_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$$

$$p(x, y|\theta) = p(y|\theta)p(x|y, \theta)$$

$$= w_y \mathcal{N}(x; \mu_y, \Sigma_y)$$

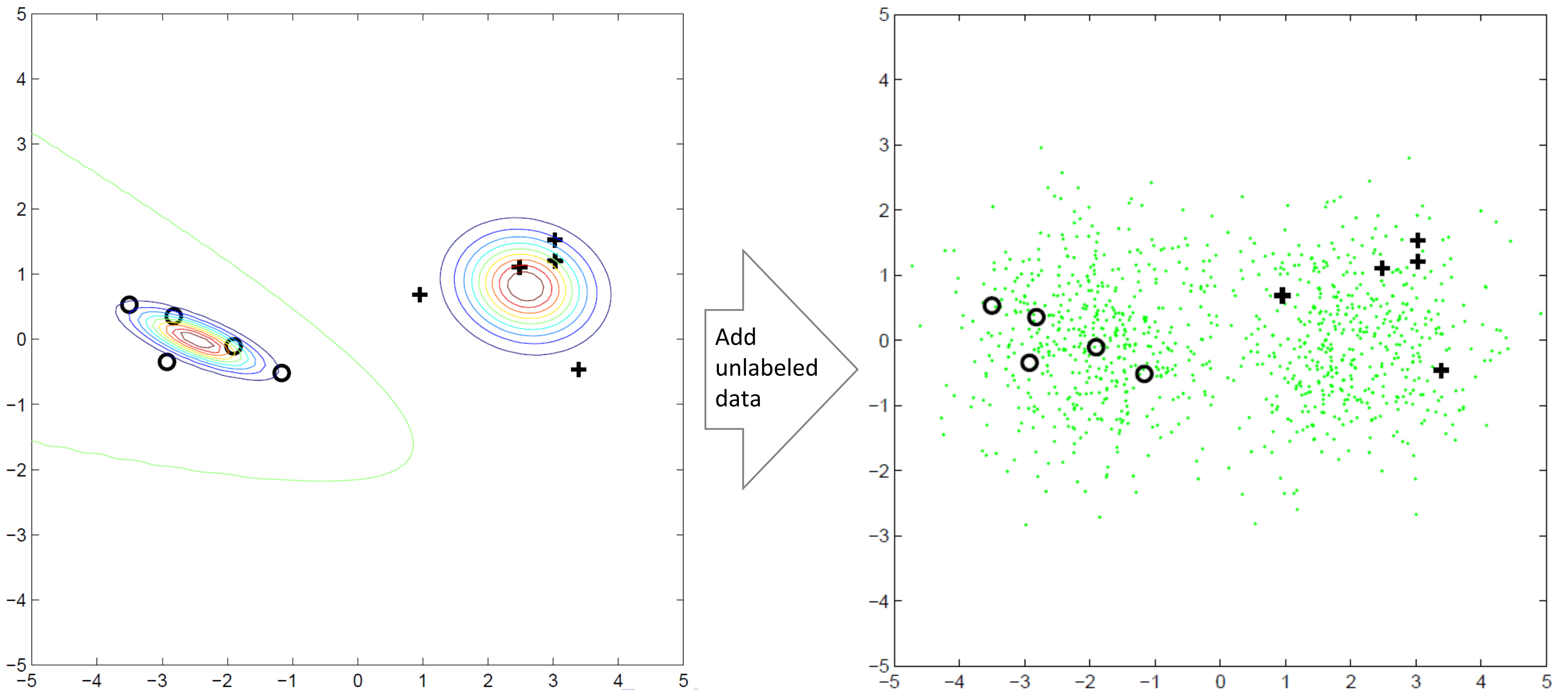
Classification :

$$p(y|x, \theta) = \frac{p(x, y|\theta)}{\sum_{y'} p(x, y'|\theta)}$$

Generative Models

Zhu (2007)

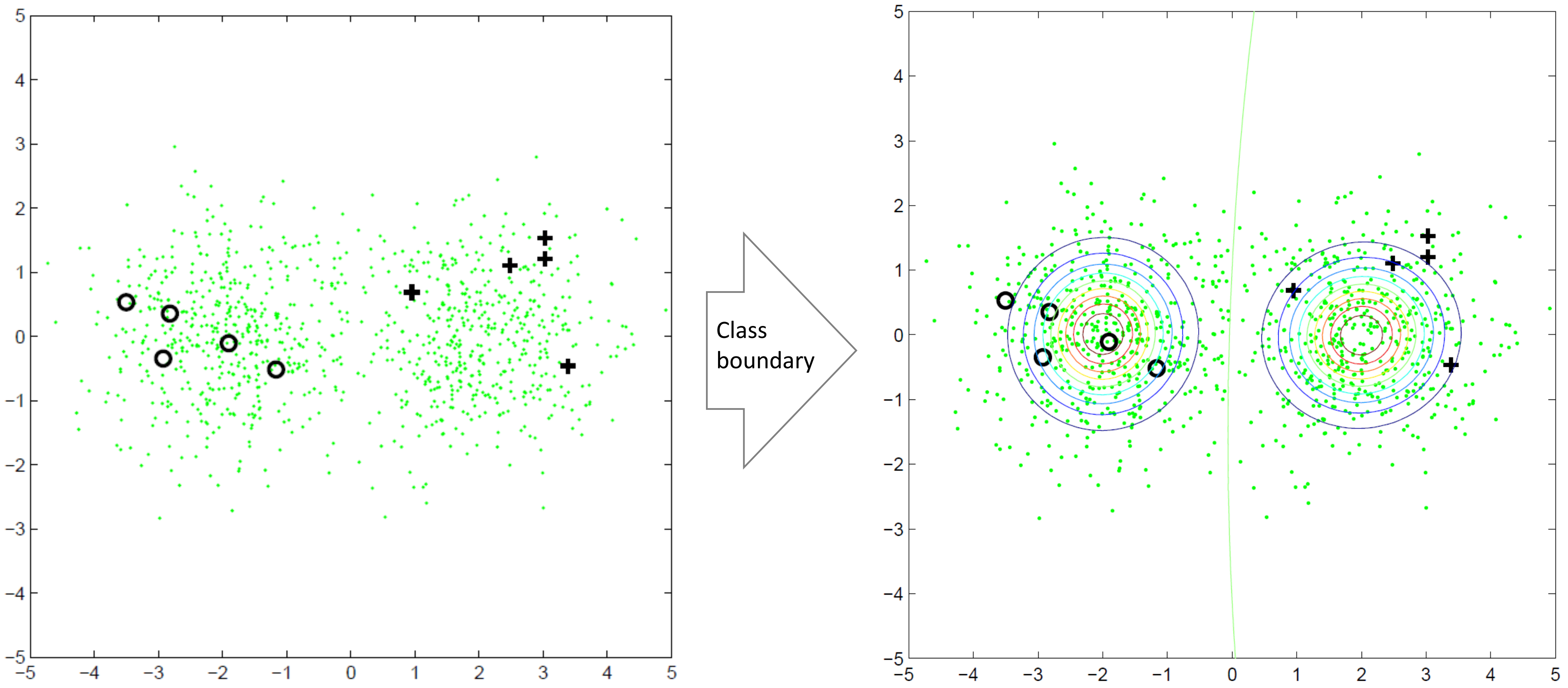
- Example
  - ✓ The most likely model, and its decision boundary:



Generative Models

Zhu (2007)

- Example
  - ✓ The most likely model, and its decision boundary:

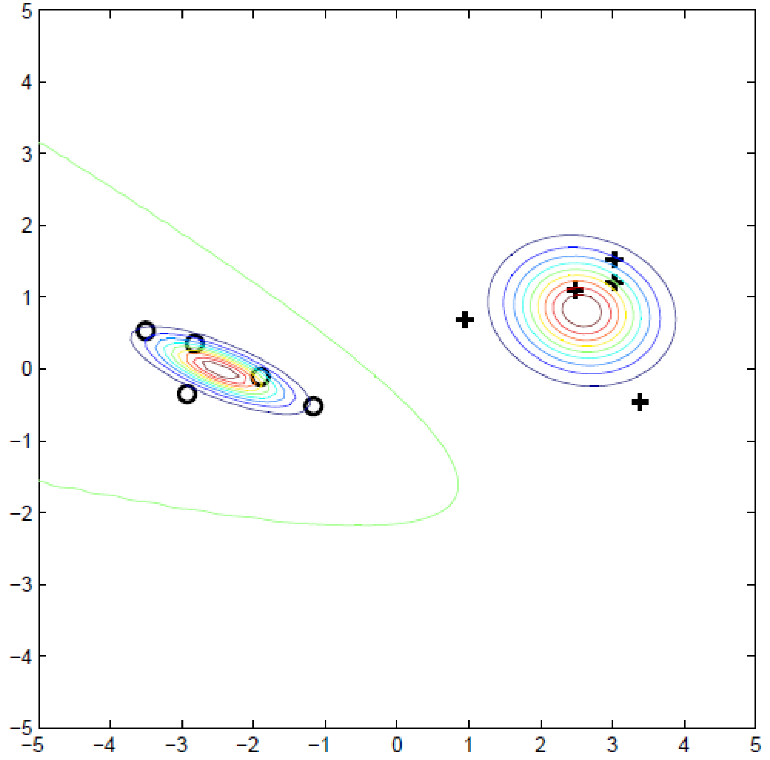


Generative Models

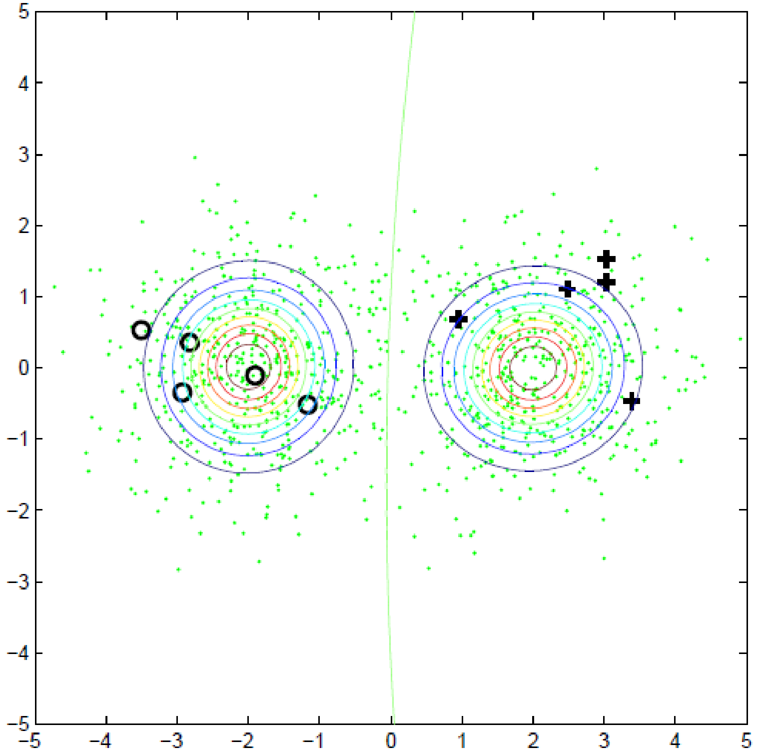
Zhu (2007)

- Example
  - ✓ They are different because they maximize different quantities

$$p(X_l, Y_l|\theta)$$



$$p(X_l, Y_l, X_u|\theta)$$



Generative Models

Fox-Roberts and Rosten (2014), Kingma et al. (2014)

- Assumption: The full generative model $p(\mathbf{X}, \mathbf{y}|\theta)$
  - ✓ Generative model for semi-supervised learning:
    - Quantity of interests:

$$p(\mathbf{X}_l, \mathbf{y}_l, \mathbf{X}_u|\theta) = \sum_{\mathbf{y}_u} p(\mathbf{X}_l, \mathbf{y}_l, \mathbf{X}_u, \mathbf{y}_u|\theta)$$

    - Find the maximum likelihood estimate (MLE) of θ, the maximum a posteriori (MAP) estimate, or be Bayesian
  - ✓ Examples of some generative models
    - Mixture of Gaussian distribution (GMM): Image classification (EM algorithm)
    - Mixture of multinomial distribution (Naïve Bayes): Text categorization (EM algorithm)
    - Hidden Markov Models (HMM): Speech recognition (Baum-Welch algorithm)

Generative Models: Gaussian Mixture Model

Zhu (2007)

- Gaussian Mixture Model
  - ✓ For simplicity, consider binary classification with GMM using MLE
    - Labeled data only

$$p(\mathbf{X}_l, \mathbf{y}_l|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta)p(\mathbf{x}_i|y_i, \theta)$$

- MLE for is θ trivial (frequency, sample mean, sample covariance)

    - Labeled and unlabeled data

$$p(\mathbf{X}_l, \mathbf{y}_l, \mathbf{X}_u|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta)p(\mathbf{x}_i|y_i, \theta) + \sum_{i=l+1}^{l+u} \log\left(\sum_{y=1}^{2} p(y|\theta)p(\mathbf{x}_i|y, \theta)\right)$$

- MLE is difficult because of hidden variables
- The Expectation-Maximization (EM) algorithm is used to find a local optimum

고려대학교 KOREA UNIVERSITY

DSBA Data Science & Business Analytics

Generative Models: Gaussian Mixture Model

Zhu (2007)

- The EM algorithm for GMM
  - ✓ Step 0: Start from MLE $\theta = \{w, \mu, \Sigma\}_{1,2}$ on $(\mathbf{X}_l, \mathbf{y}_l)$, repeat
  - ✓ Step 1: The E-Step: compute the expected label for all $\mathbf{x} \in \mathbf{X}_u$

$$p(y|\mathbf{x}, \theta) = \frac{p(\mathbf{x}, y|\theta)}{\sum_{y'} p(\mathbf{x}, y'|\theta)}$$

    - label $p(y = 1|\mathbf{x}, \theta)$-fraction of **x** with class 1
    - label $p(y = 2|\mathbf{x}, \theta)$-fraction of **x** with class 2
  - ✓ Step 2: The M-Step: update MLE θ with (now labeled) $\mathbf{X}_u$
    - w_c: proportion of class c
    - μ_c: sample mean of class c
    - Σ_c: sample covariance of class c
- Can be viewed as a special form of self-training

Generative Models: Gaussian Mixture Model

Zhu (2007)

- The EM algorithm in general
  - ✓ Set up:
    - Obtain data $\mathcal{D} = (\mathbf{X}_l, \mathbf{y}_l, \mathbf{X}_u)$
    - hidden data $\mathcal{H} = \mathbf{y}_u$

$$p(\mathcal{D}|\theta) = \sum_{\mathcal{H}} p(\mathcal{D}, \mathcal{H}|\theta)$$

  - ✓ Goal: find θ to maximize $p(\mathcal{D}|\theta)$
  - ✓ Properties
    - EM starts from an arbitrary θ_0
    - The E-step: $q(\mathcal{H}) = p(\mathcal{H}|\mathcal{D}, \theta)$
    - The M-step: maximize $\sum_{q(\mathcal{H})} \log p(\mathcal{D}, \mathcal{H}|\theta)$
    - EM iteratively improves $p(\mathcal{D}|\theta)$
    - EM converges to a local maximum of θ

Generative Models: Gaussian Mixture Model

Zhu (2007)

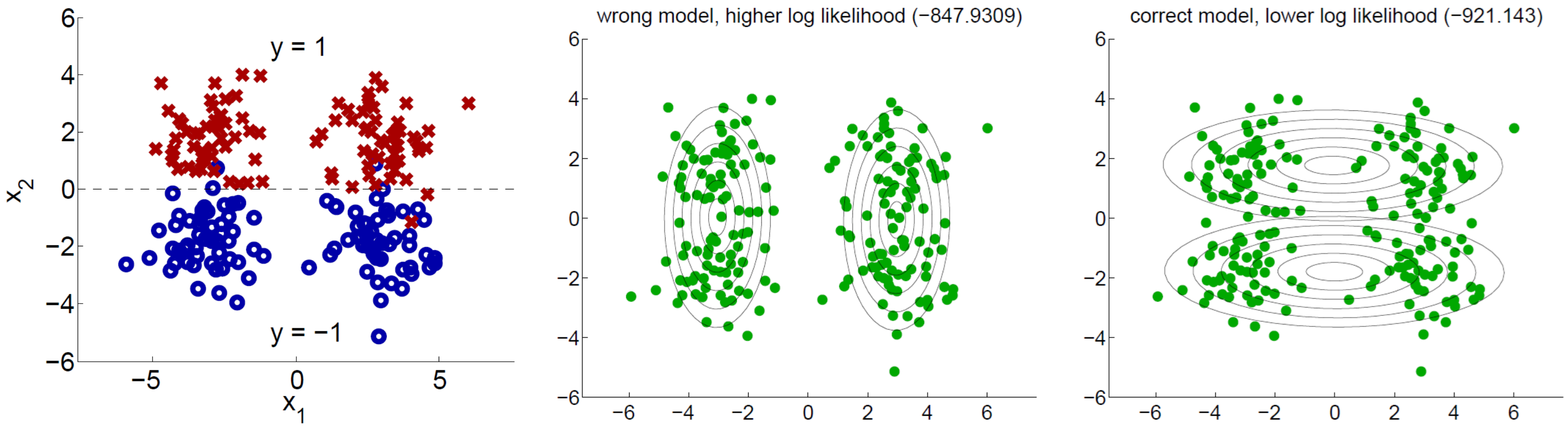
- The EM algorithm in general
  - ✓ Key is to maximize the posterior probability
  - ✓ EM is just one way to maximize it; other ways to find parameters are possible (variational approximation, direct optimization, etc.)

- Advantages
  - ✓ Clear, well-studied probabilistic framework
  - ✓ Can be extremely effective, if the model is close to correct

Generative Models: Gaussian Mixture Model

Zhu (2007)

- Disadvantages
  - ✓ Often difficult to verify the correctness of the model
  - ✓ Model identifiability, local optima of the EM algorithm
  - ✓ Unlabeled data may hurt if generative model is wrong



Semi-supervised Learning with Generative Models

- Recent Advances

Semi-supervised Learning with Generative Models

- Variational Autoencoder
- GAN or Adversarial Training
- Consistency Training

2014
- Semi-supervised Learning with Deep Generative Models (Kingma et al., 140620, NIPS 2014)

2015
- Semi-supervised Learning w Ladder Networks (Rasmus et al., 150709, NIPS

- Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks (Springenberg, 151119, ICLR2016)

2016
- Temporal Ensembling for Semi-Supervised Learning (Laine and Aila, 161007, ICLR 2017)
- Semi-Supervised Learning with Generative Adversarial Networks (Odena, 160605, ICML 2016)

2017
- Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning (Miyato et al., 170413, IEEE TPAMI 2019)
- Good Semi-supervised Learning that Requires a Bad GAN (Dai et al., 170527, NIPS 2017)
- Learning Disentangled Representations with Semi-Supervised Deep Generative Models (Siddharth et al., 170601, NIPS 2017)

2018
- Realistic Evaluation of Deep Semi-Supervised Learning Algorithms (Oliver et al., 080424, NeuralPS 2018)

2019
- Label Propagation for Deep Semi-supervised Learning (Iscen et al., 190409, CVPR 2019)
- Unsupervised Data Augmentation for Consistency Training (Xie et al., 190429, 2019)
- MixMatch: A Holistic App o Semi-Supervised Learnin (Berthelot et al., 190506, N 2019)

- Remixmatch: Semi-super learning with distribution ent and augmentation ancho (Berthelot et al., 191121, 2019)

- S4L: Self-Supervised Semi-Supervised Learning (Zhai et al, 190509, ICCV 2019)

2020
- FixMatch: Simplifying Semi-Super Learning with Consistency and Co e (Kurakin et al., 200121, NeurIPS 20

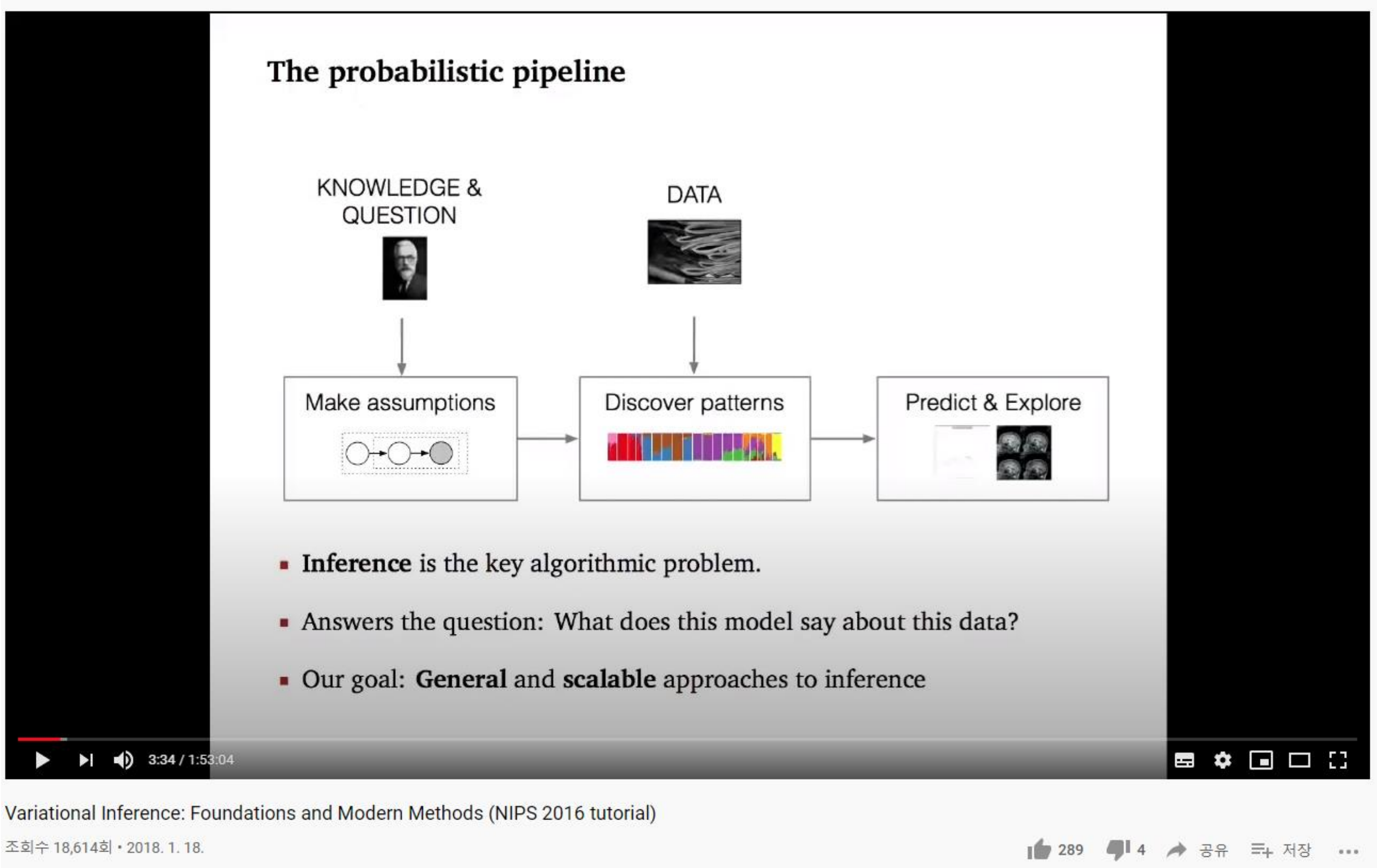
고려대학교 KOREA UNIVERSITY

DSBA Data Science & Business Analytics

Deep Generative Models

Kingma et al. (2014)

- Variational Inference
  - ✓ David Blei, Variational Inference: Foundations and Modern Methods (NIPS 2016 tutorial)
    - https://youtu.be/ogdv_6dbvVQ



Deep Generative Models

Kingma et al. (2014)

- Variational Lower Bound
  - ✓ We have some data $\boldsymbol{X}$, a generative probability model $P(\boldsymbol{X}|\theta)$ that shows us how to randomly sample (e.g., generate) data points that follow distribution of $\boldsymbol{X}$, assuming we know the "magic values" of the θ parameters

$$p(\theta|\mathbf{X}) = \frac{p(\mathbf{x}|\theta)p(\theta)}{p(\mathbf{X})} = \frac{p(\mathbf{x}|\theta)p(\theta)}{\int_{-\infty}^{\infty} p(\mathbf{x}|\theta)p(\theta)d\theta}$$

$$\text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}}$$

  - ✓ The goal is to find the posterior $P(\theta|\boldsymbol{X})$, that tells us the distribution of the θ parameters
    - Sometimes is the end goal
    - Sometimes we want the parameters to generate some new data points using $P(\boldsymbol{X}|\theta)$
  - ✓ The problem is intractable because of its denominator

Deep Generative Models

Kingma et al. (2014)

- The solution: Approximation
  - ✓ Approximate $P(\theta|\boldsymbol{X})$ by another function $Q(\theta|\boldsymbol{X})$
    - Solving for Q is relatively fast because we can assume a particular shape for $Q(\theta|\boldsymbol{X})$ and turn the inference problem (finding $P(\theta|\boldsymbol{X})$) into an optimization problem (finding Q)
    - $Q(\theta|\boldsymbol{X})$ should be as close as possible to $P(\theta|\boldsymbol{X})$
  - ✓ In terms of "closeness", the standard way of measuring it is to use KL divergence

$$
\begin{aligned}
D_{KL}(Q||P) &= \int_{-\infty}^{\infty} q(\theta|\mathbf{X})\log\frac{q(\theta|\mathbf{X})}{p(\theta|\mathbf{X})}d\theta \\
&= \int_{-\infty}^{\infty} q(\theta|\mathbf{X})\log\frac{q(\theta|\mathbf{X})}{p(\theta,\mathbf{X})}d\theta + \int_{-\infty}^{\infty} q(\theta|\mathbf{X})\log p(\mathbf{X})d\theta \\
&= \int_{-\infty}^{\infty} q(\theta|\mathbf{X})\log\frac{q(\theta|\mathbf{X})}{p(\theta,\mathbf{X})}d\theta + \log p(\mathbf{X}) \\
&= E_q\left[\log\frac{q(\theta|\mathbf{X})}{p(\theta,\mathbf{X})}\right] + \log p(\mathbf{X})
\end{aligned}
$$

Deep Generative Models

Kingma et al. (2014)

- Evidence Lower Bound (ELBO):
  - ✓ For multiple data points, we can sum over them
  - ✓ ELBO is a lower bound on the evidence, i.e., a lower bound on the probability of our data occurring given our model
  - ✓ Maximizing the ELBO is equivalent to minimizing the KL divergence
  - ✓ The first two terms try to maximize the MAP estimate (likelihood + prior)
  - ✓ The last term tries to ensure Q is diffuse

$$\log p(\mathbf{X}) = -E_q\left[\log \frac{q(\theta|\mathbf{X})}{p(\theta, \mathbf{X})}\right] + D_{KL}(Q||P)$$

$$\log p(\mathbf{X}) \geq -E_q\left[\log \frac{q(\theta|\mathbf{X})}{p(\theta, \mathbf{X})}\right]$$

$$= E_q[\log p(\theta, \mathbf{X}) - \log q(\theta|\mathbf{X})]$$

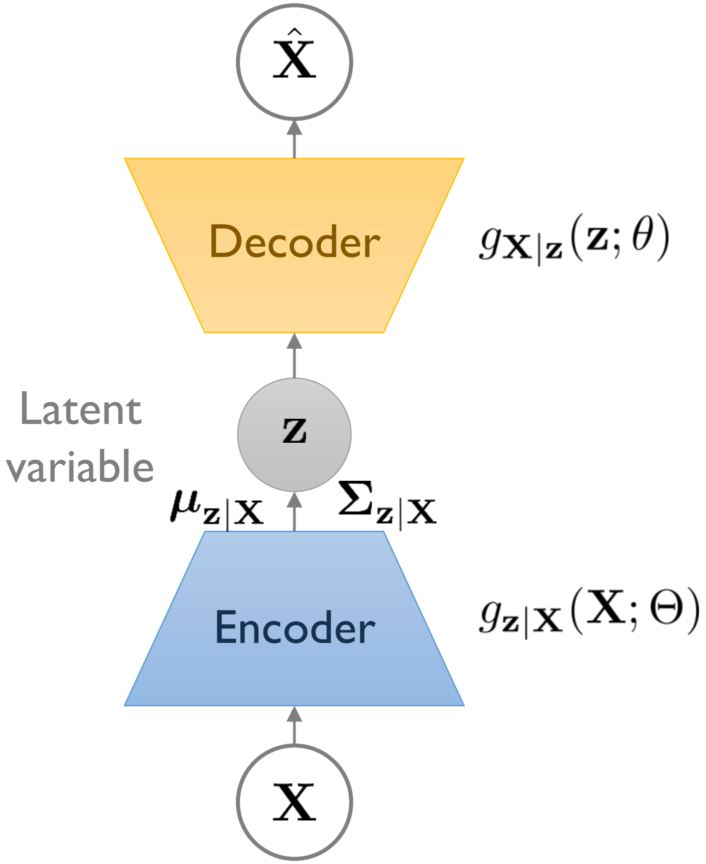
$$= E_q[\log p(\mathbf{X}|\theta) + \log p(\theta) - \log q(\theta|\mathbf{X})]$$

$$= E_q[\text{likelohood} + \text{prior} - \text{approx.posterior}]$$

Deep Generative Models

Kingma et al. (2014)

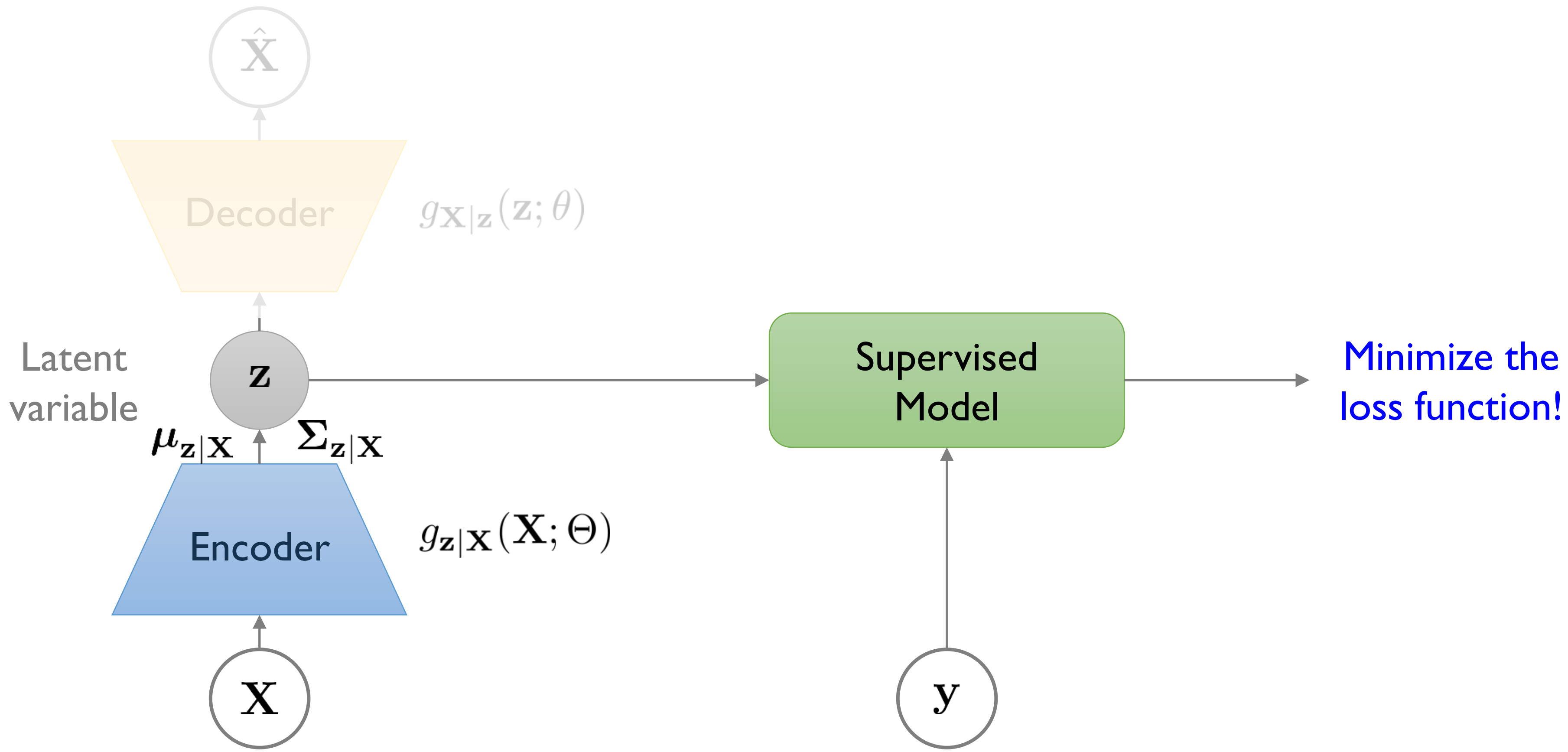
- A Vanilla VAE for Semi-supervised Learning (M1 Model)
  - ✓ Phase 1: train a variational auto-encoder (VAE) based on both labeled and unlabeled data



Deep Generative Models

Kingma et al. (2014)

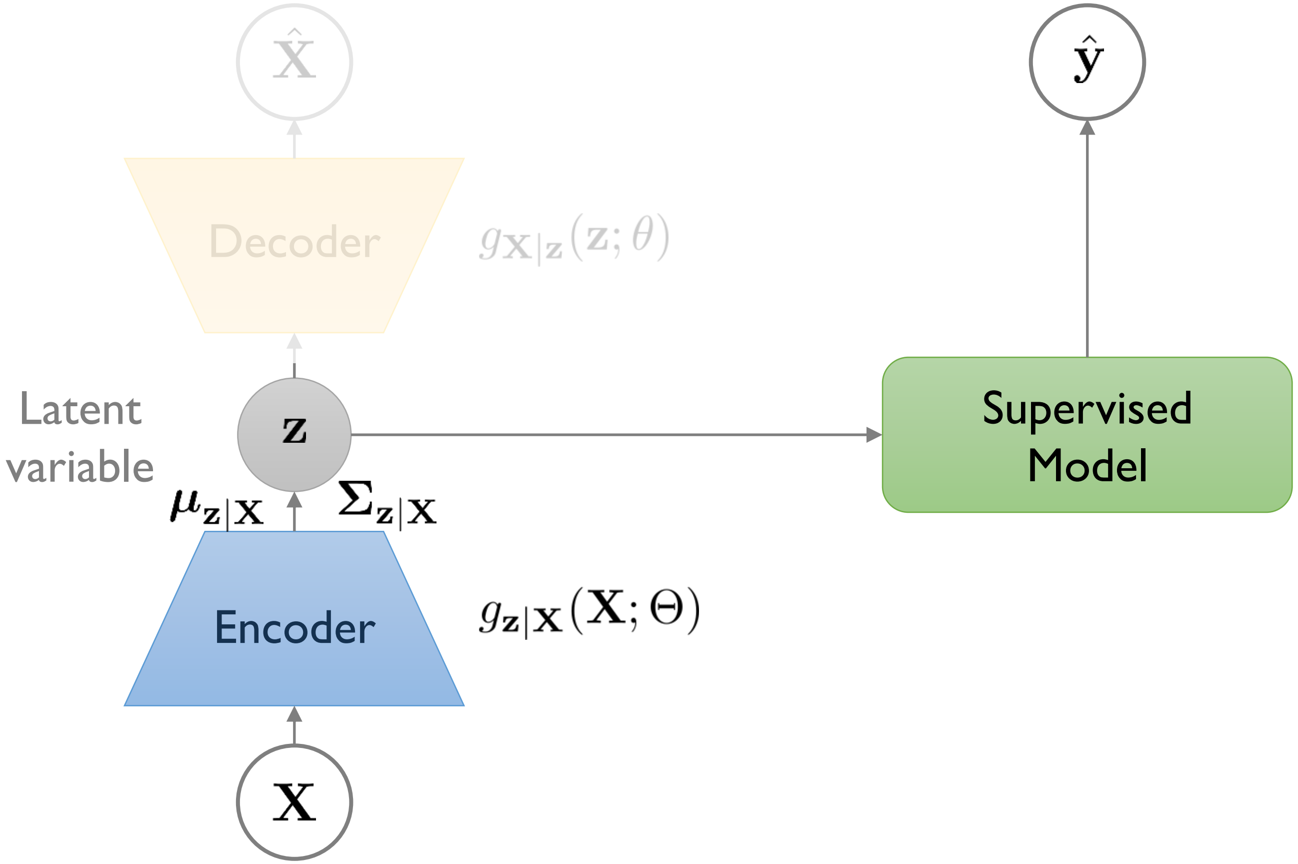
- A Vanilla VAE for Semi-supervised Learning (M1 Model)
  - ✓ Phase 2: Solve a standard supervised learning problem on the labeled data using $(\mathbf{Z}, \mathbf{y})$ pairs



Deep Generative Models

Kingma et al. (2014)

- A Vanilla VAE for Semi-supervised Learning (M1 Model)
  - ✓ Phase 3: Inference for unlabeled data or new test data

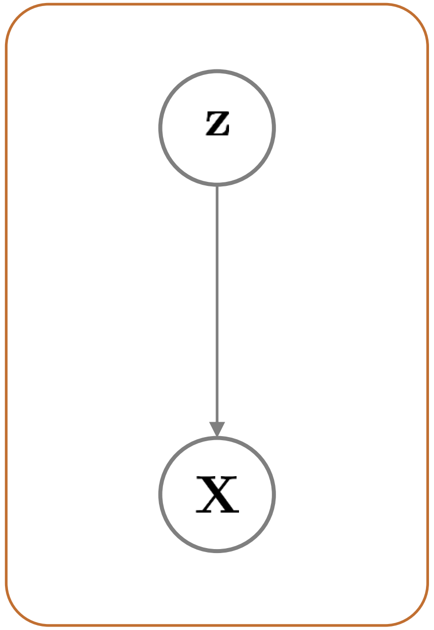


Deep Generative Models

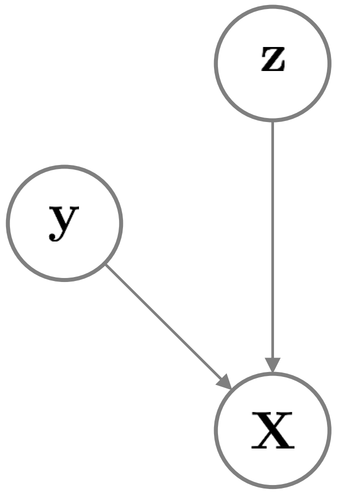
Kingma et al. (2014)

- Extending the VAE for Semi-supervised Learning (M2 Model)
  - ✓ M1 model basically ignored the labeled data when training the VAE model
  - ✓ M2 model explicitly takes in into account

M1 Model



M2 Model



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

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

$$p(y|\pi) = \mathrm{Cat}(y|\pi)$$

$$p(\pi) = \mathrm{SymDir}(\alpha)$$

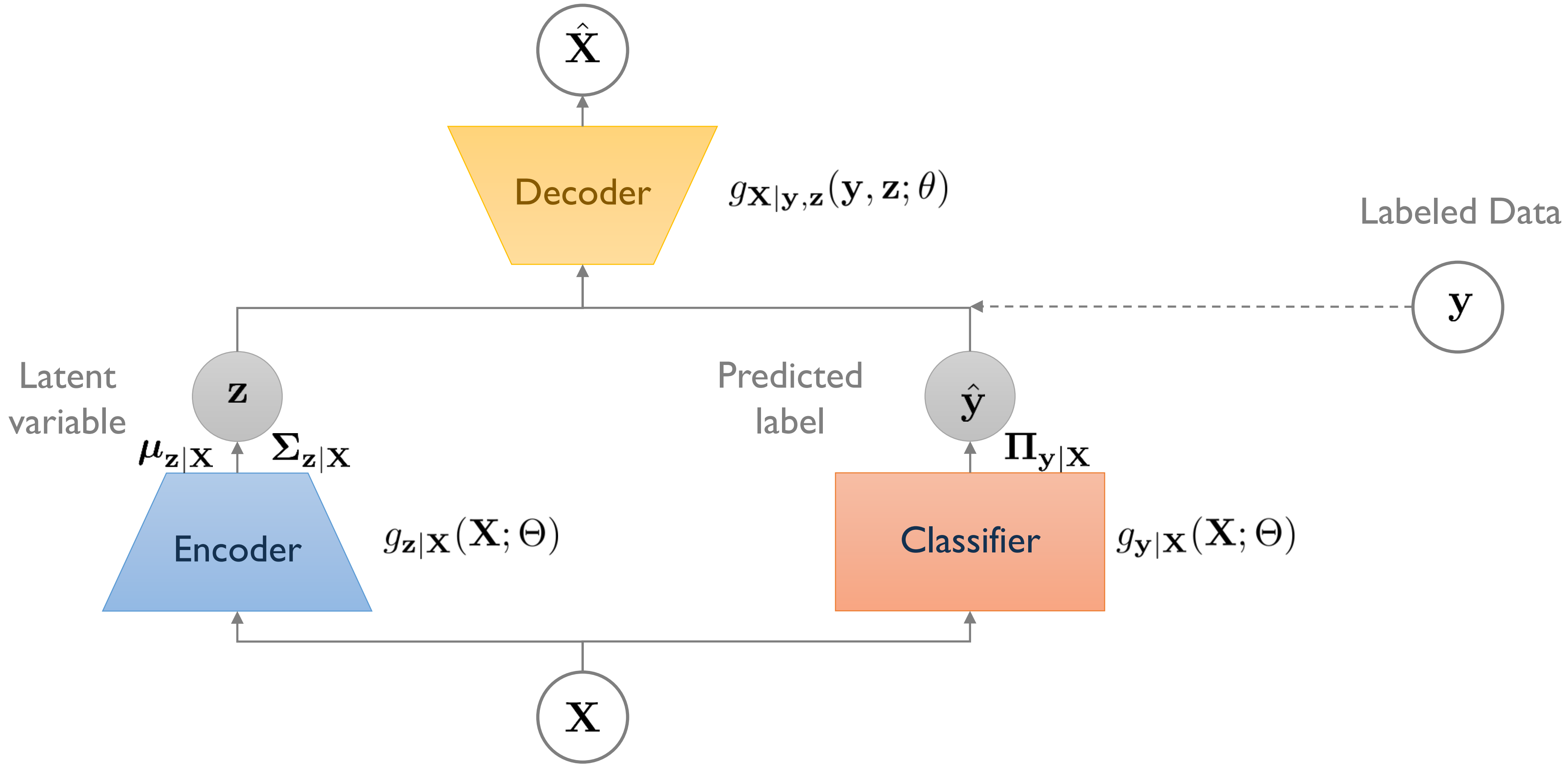
고려대학교 KOREA UNIVERSITY

DSBA Data Science & Business Analytics

Deep Generative Models

Kingma et al. (2014)

- Extending the VAE for Semi-supervised Learning (M2 Model)



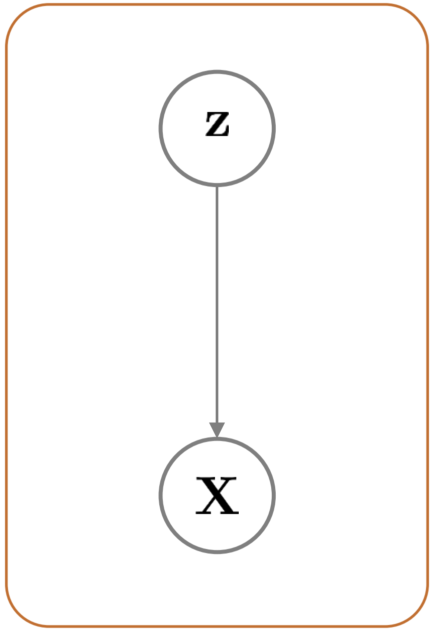
Deep Generative Models

Kingma et al. (2014)

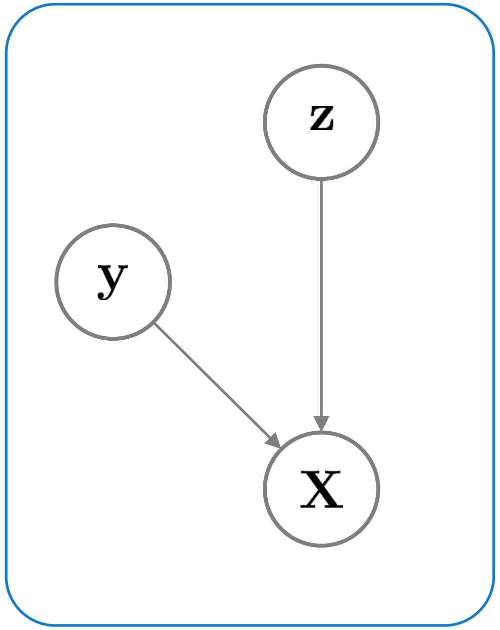
- Stack of M1 and M2
  - ✓ Train generative semi-supervised model (M2) on unsupervised features z_1 from latent feature model (M1)

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

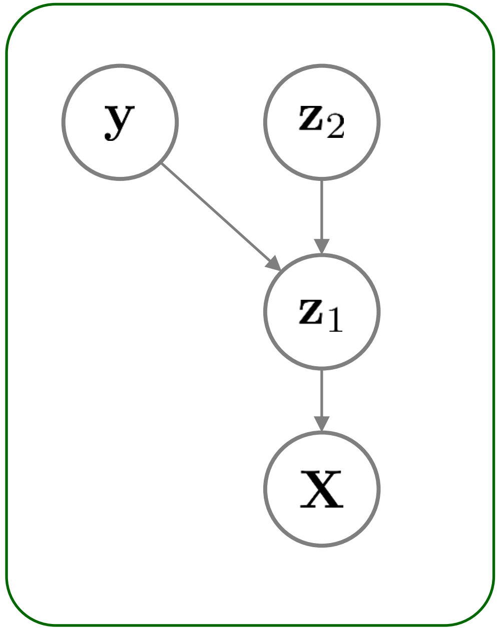
M1 Model

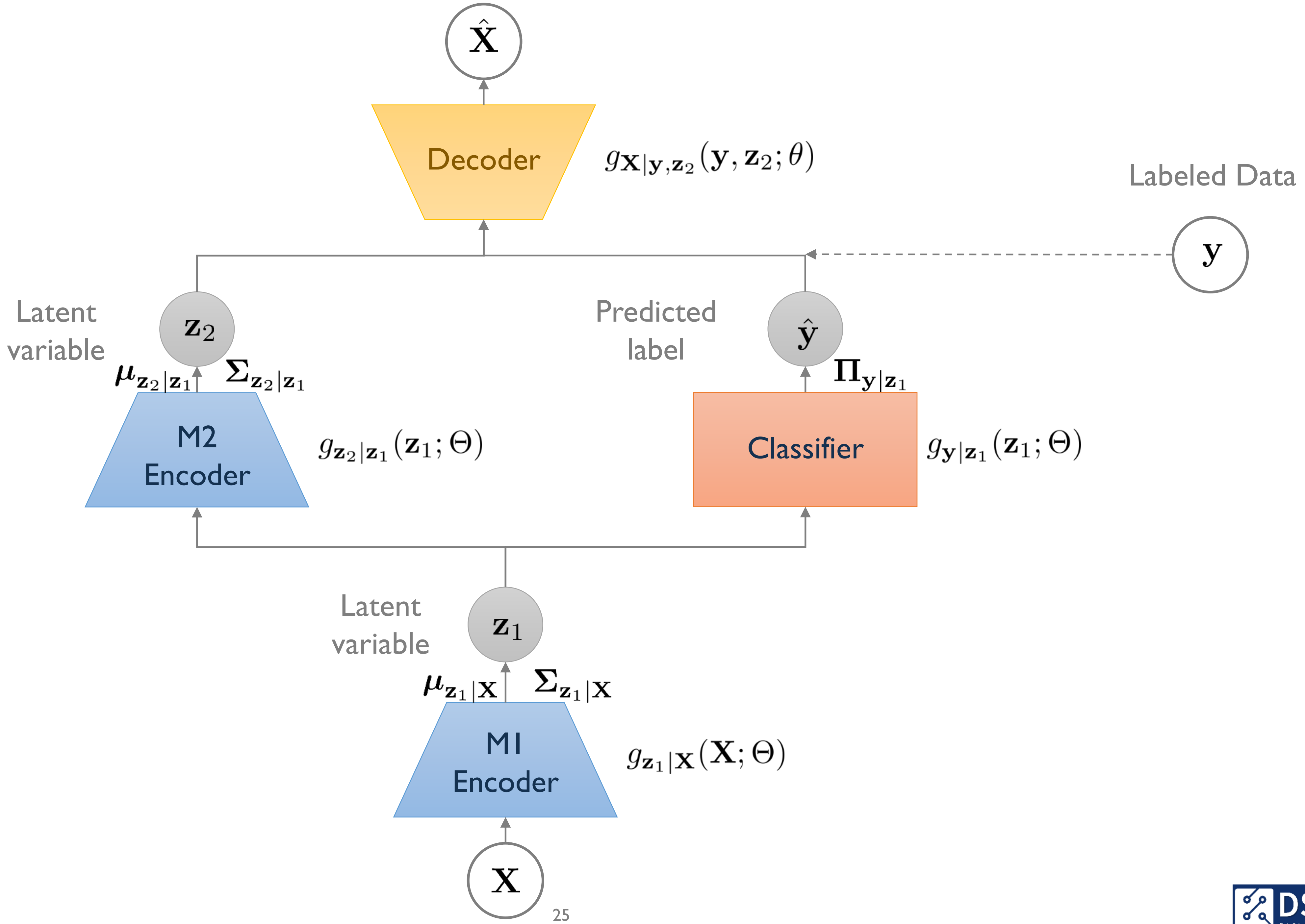


M2 Model



M1+M2 Model





$\hat{\mathbf{X}}$
Decoder
$g_{\mathbf{X}|\mathbf{y},\mathbf{z}_2}(\mathbf{y}, \mathbf{z}_2; \theta)$
Labeled Data
$\mathbf{y}$
Latent variable
$\mathbf{z}_2$
Predicted label
$\hat{\mathbf{y}}$
$\boldsymbol{\mu}_{\mathbf{z}_2|\mathbf{z}_1}$
$\boldsymbol{\Sigma}_{\mathbf{z}_2|\mathbf{z}_1}$
$\boldsymbol{\Pi}_{\mathbf{y}|\mathbf{z}_1}$
M2 Encoder
$g_{\mathbf{z}_2|\mathbf{z}_1}(\mathbf{z}_1; \Theta)$
Classifier
$g_{\mathbf{y}|\mathbf{z}_1}(\mathbf{z}_1; \Theta)$
Latent variable
$\mathbf{z}_1$
$\boldsymbol{\mu}_{\mathbf{z}_1|\mathbf{X}}$
$\boldsymbol{\Sigma}_{\mathbf{z}_1|\mathbf{X}}$
M1 Encoder
$g_{\mathbf{z}_1|\mathbf{X}}(\mathbf{X}; \Theta)$
$\mathbf{X}$

Deep Generative Models

Kingma et al. (2014)

- Performance (from paper)

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)

- Performance (implemented, https://bjlkeng.github.io/posts/semi-supervised-learning-with-variational-autoencoders/)

Model	**N=100**	**N=500**	**N=1000**	**N=2000**	**N=5000**
PCA + SVM	0.692	0.871	0.891	0.911	0.929
CNN	0.262	0.921	0.934	0.955	0.978
M1	0.628	0.885	0.905	0.921	0.933
M2	•	•	0.975	•	•

Table 1: MNIST Results

Model	**N=1000**	**N=2000**	**N=5000**	**N=10000**	**N=25000**
CNN	0.433	0.4844	0.610	0.673	0.767
Inception	0.661	0.684	0.728	0.751	0.773
PCA + SVM	0.356	0.384	0.420	0.446	0.482
M1	0.321	0.362	0.375	0.389	0.409
M2	0.420	•	•	•	•

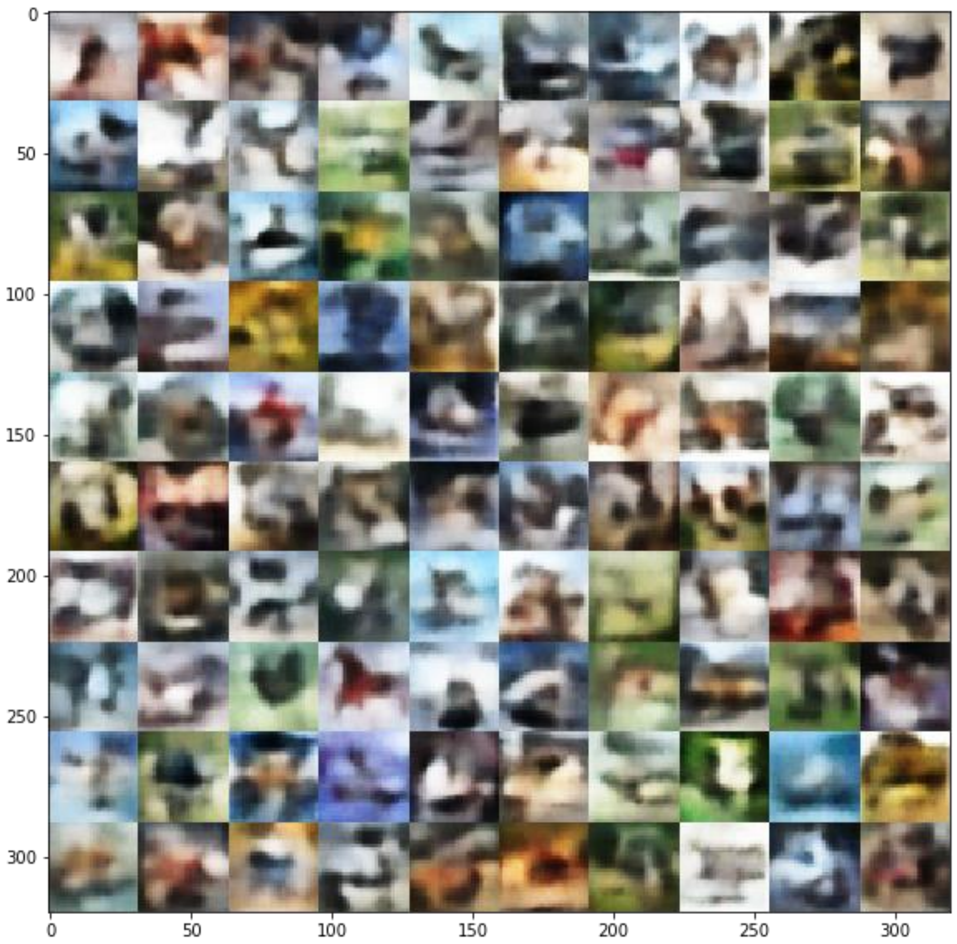
Table 2: CIFAR10 Results

Deep Generative Models

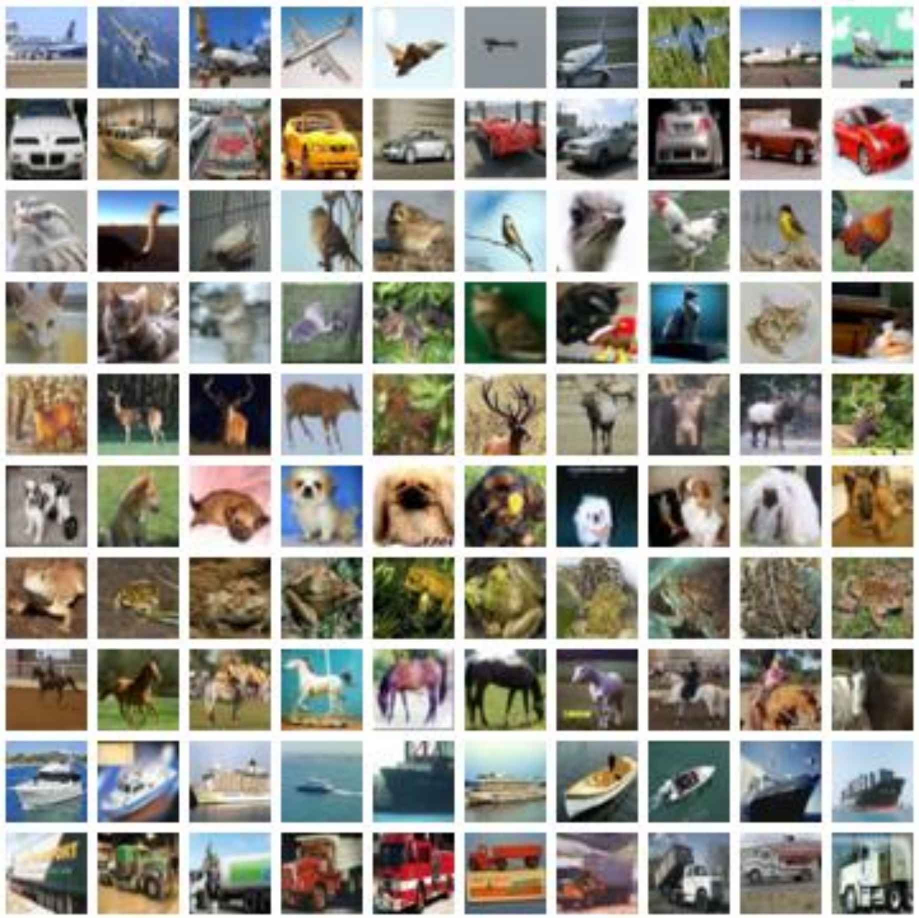
Kingma et al. (2014)

- Images generated from M2 VAE model trained on CIFAR data

Generated images



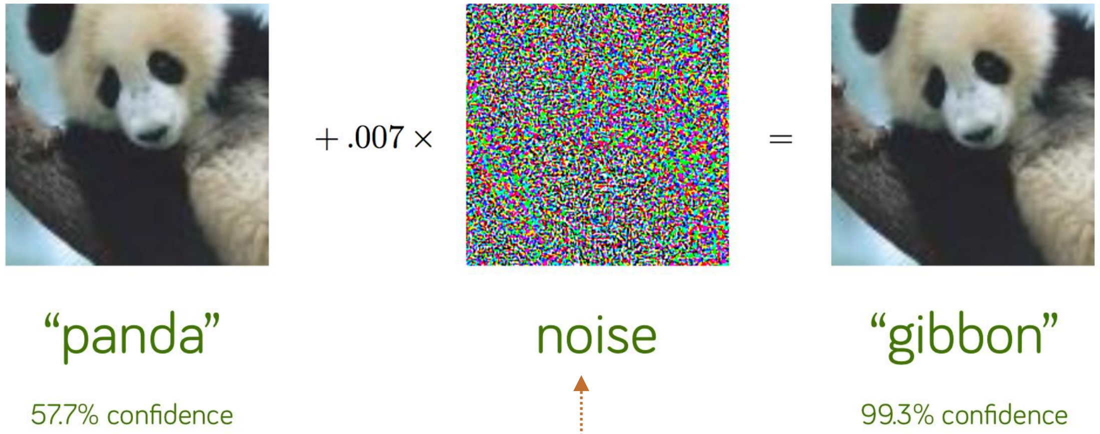
Original images



Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

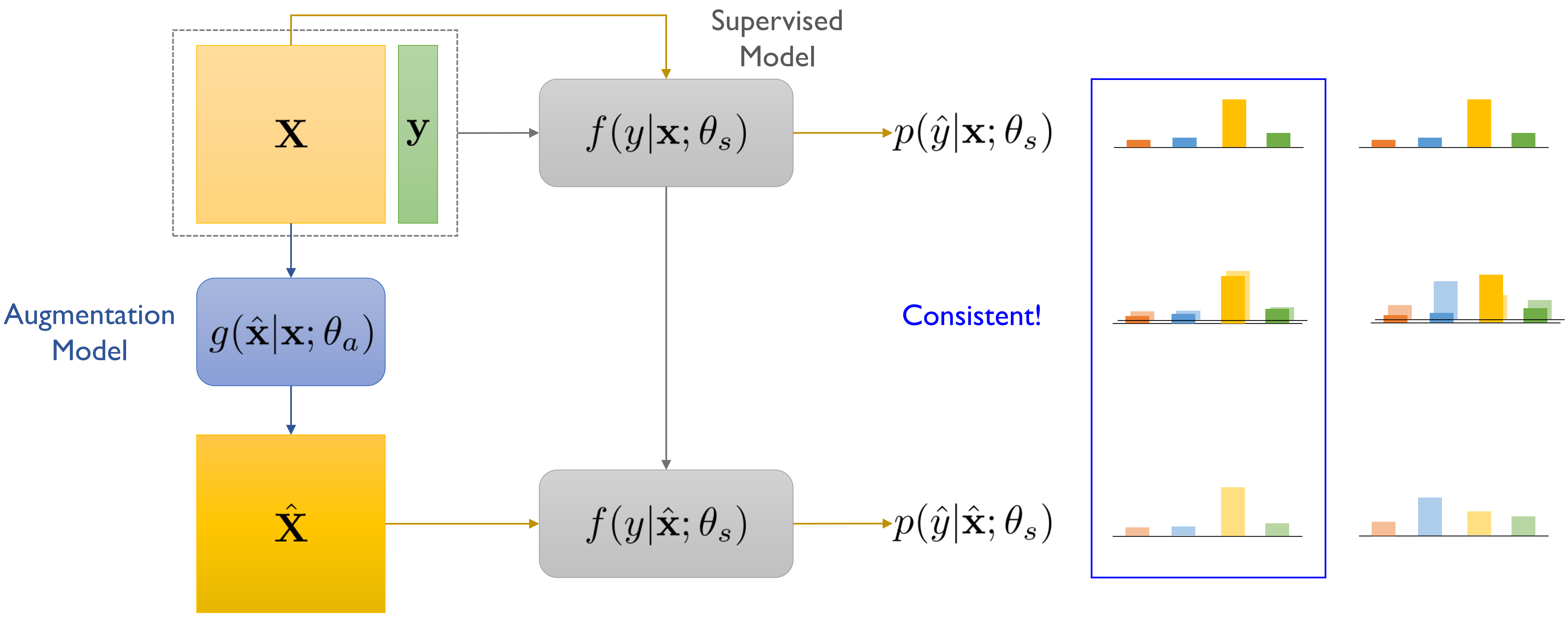
- A good model should be robust to any small change in an input example or hidden states



- Consistency training: Simply regularize model predictions to be invariant to small noise applied to either input examples or hidden states
- UDA investigate the role of noise injection in consistency training and observe that advanced data augmentation methods perform well in SSL.

Unsupervised Data Augmentation for Consistency Training

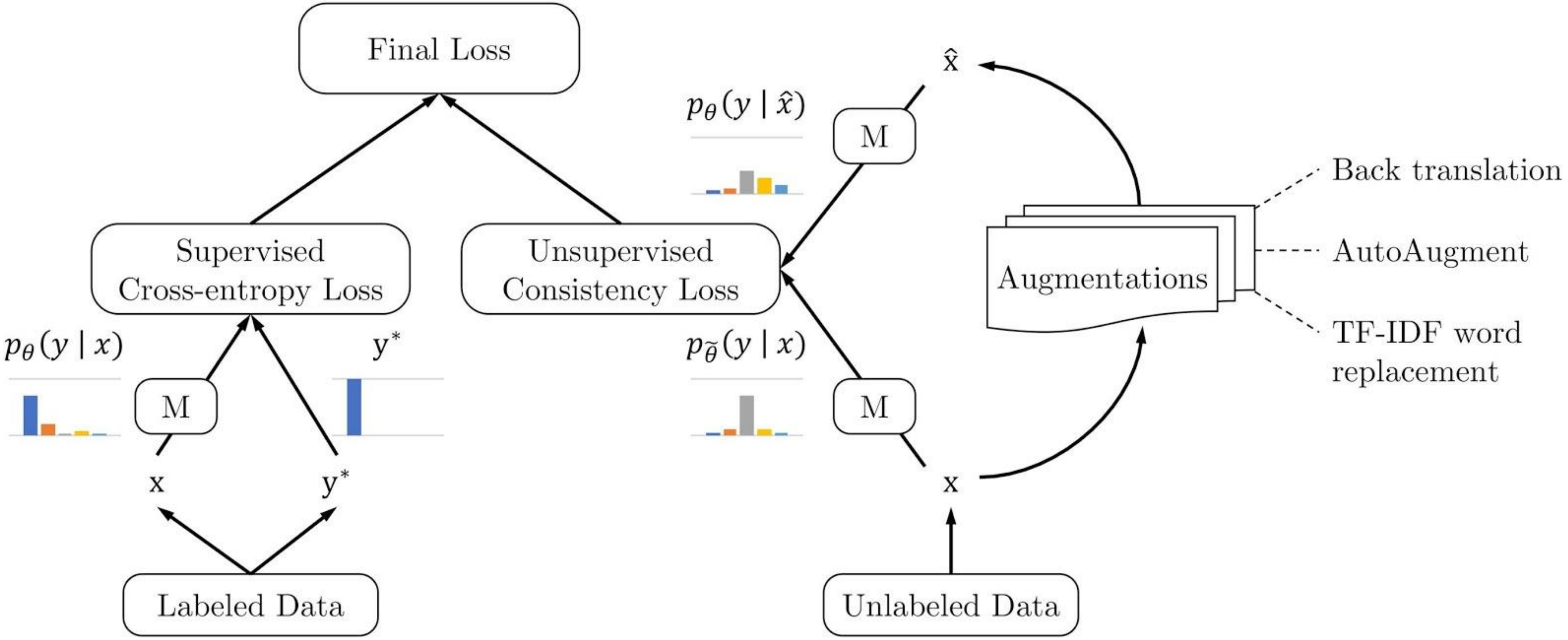
- Consistency Training



Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

- UDA at a glance

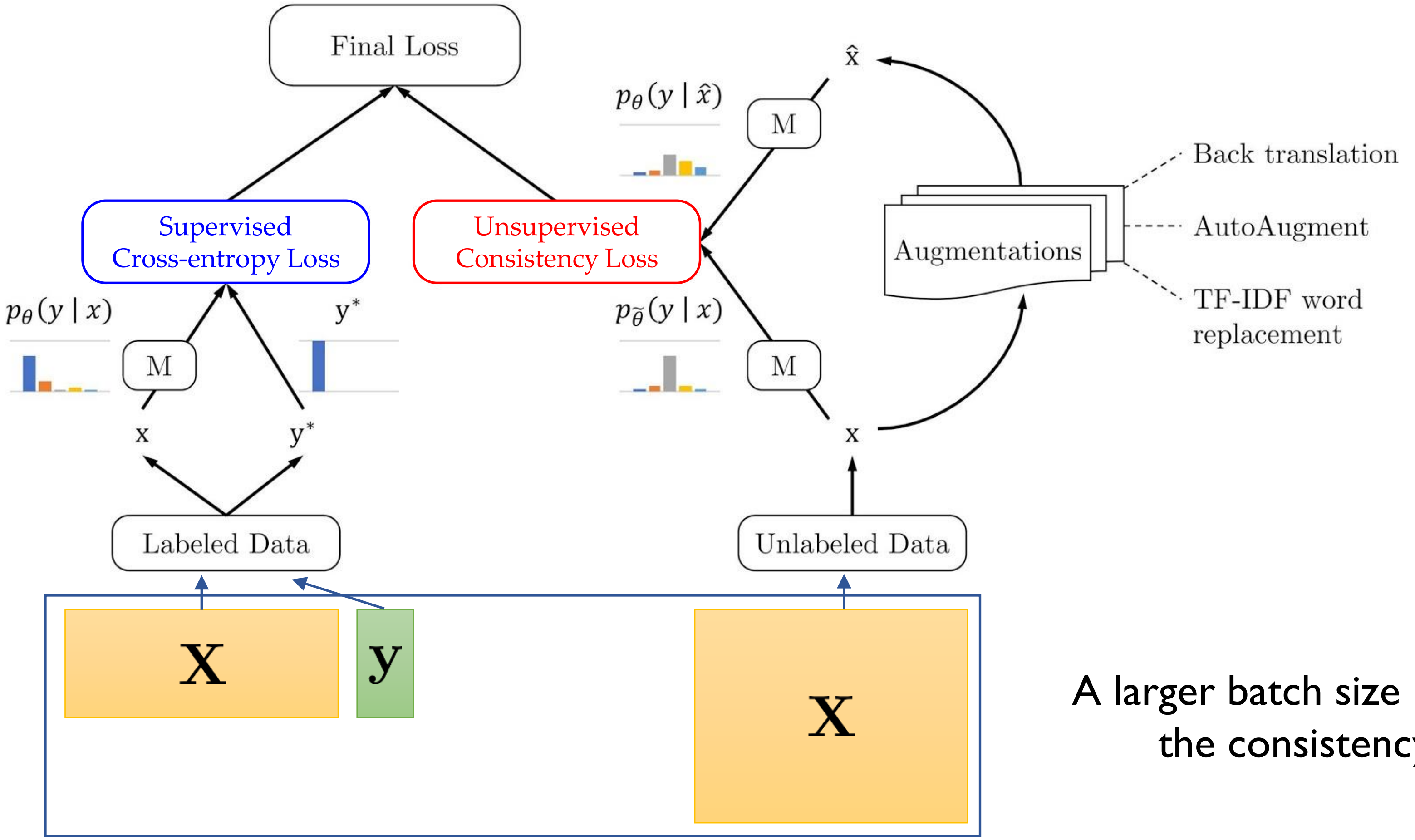


Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

- Training Objective

$$\min_{\theta} \mathcal{J}(\theta) = \mathbb{E}_{x_1 \sim p_L(x)}[-\log p_\theta(f^*(x_1)|x_1)] + \lambda \times \mathbb{E}_{x_2 \sim p_U(x)} \mathbb{E}_{\hat{x} \sim q(\hat{x}|x_2)}[\mathrm{CE}(p_{\tilde{\theta}}(y|x_2)||p_\theta(y|\hat{x}))]$$

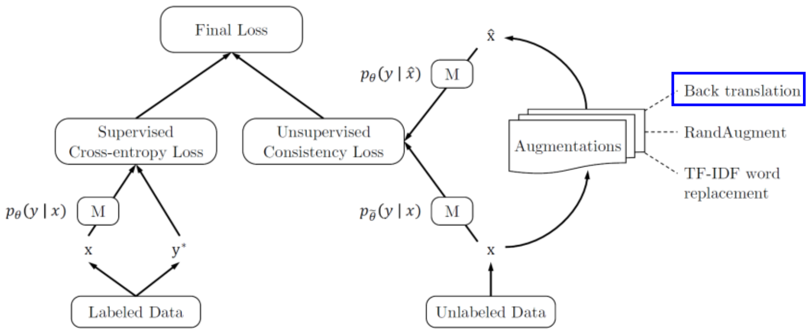


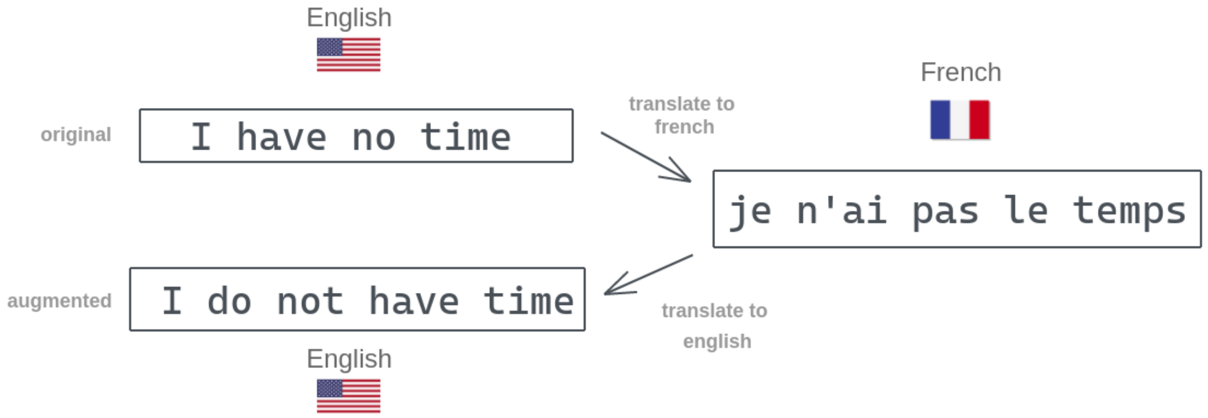
A larger batch size is used for the consistency loss

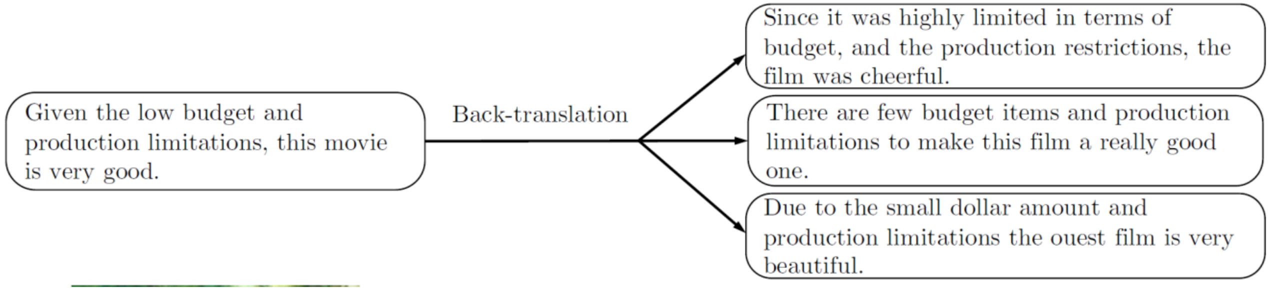
Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

- Augmentation Methods
  - ✓ Back-translation for NLP
    - Train the Model 1 that translates Language A to Language B
    - Train the Model 2 that translates Language B to Language A
    - Augment the Data by applying the source text to Model A and Model B sequentially



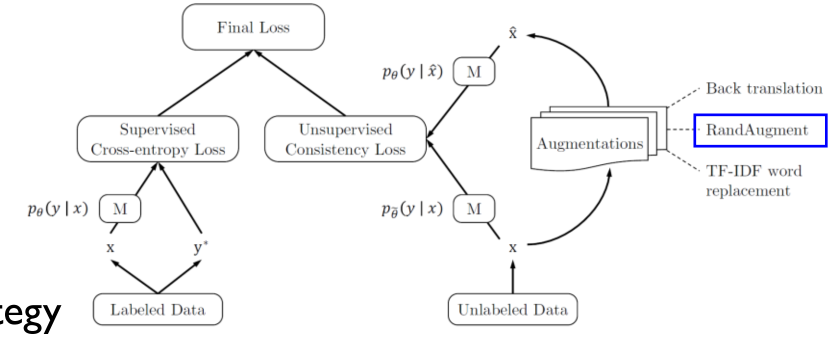




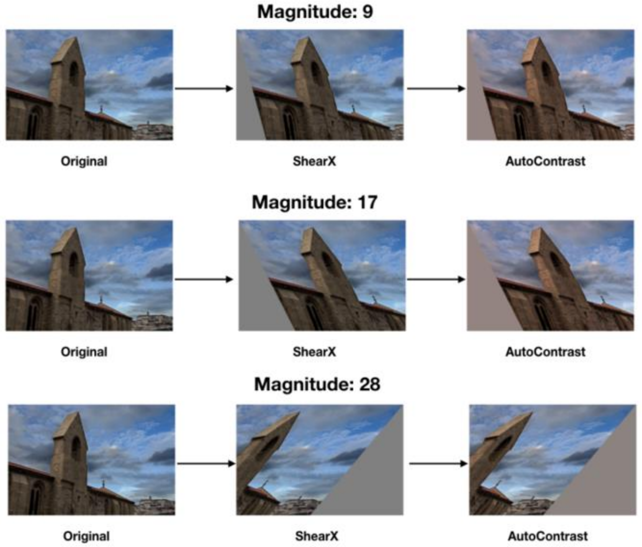
https://amitness.com/2020/02/back-translation-in-google-sheets/

Unsupervised Data Augmentation for Consistency Training

Cubuk et al. (2020)



- Augmentation Methods
  - ✓ RandAugment for Image Classification
    - AutoAugment: uses a search method to combine all image processing transformations in the Python Image Library (PIL) to find a good augmentation strategy
    - RandAugment: does not use search but instead uniformly sample from the same set of augmentation transformations in PIL (Simpler and requires no labeled data as there is no nood to search for optimal policies

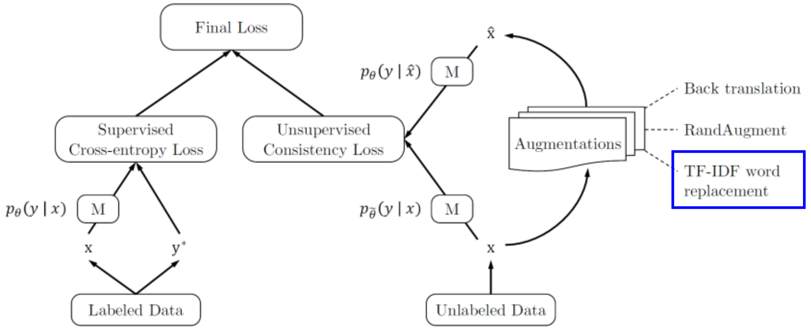



```
transforms = [
'Identity', 'AutoContrast', 'Equalize',
'Rotate', 'Solarize', 'Color', 'Posterize',
'Contrast', 'Brightness', 'Sharpness',
'ShearX', 'ShearY', 'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to
       apply sequentially.
    M: Magnitude for all the transformations.
"""

  sampled_ops = np.random.choice(transforms, N)
  return [(op, M) for op in sampled_ops]
```

	baseline	PBA	Fast AA	AA	RA
CIFAR-10					
Wide-ResNet-28-2	94.9	-	-	**95.9**	95.8
Wide-ResNet-28-10	96.1	**97.4**	97.3	**97.4**	97.3
Shake-Shake	97.1	**98.0**	**98.0**	**98.0**	**98.0**
PyramidNet	97.3	**98.5**	98.3	**98.5**	**98.5**
CIFAR-100					
Wide-ResNet-28-2	75.4	-	-	**78.5**	78.3
Wide-ResNet-28-10	81.2	**83.3**	82.7	82.9	**83.3**
SVHN (core set)					
Wide-ResNet-28-2	96.7	-	-	98.0	**98.3**
Wide-ResNet-28-10	96.9	-	-	98.1	**98.3**
SVHN					
Wide-ResNet-28-2	98.2	-	-	**98.7**	**98.7**
Wide-ResNet-28-10	98.5	98.9	98.8	98.9	**99.0**

Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)



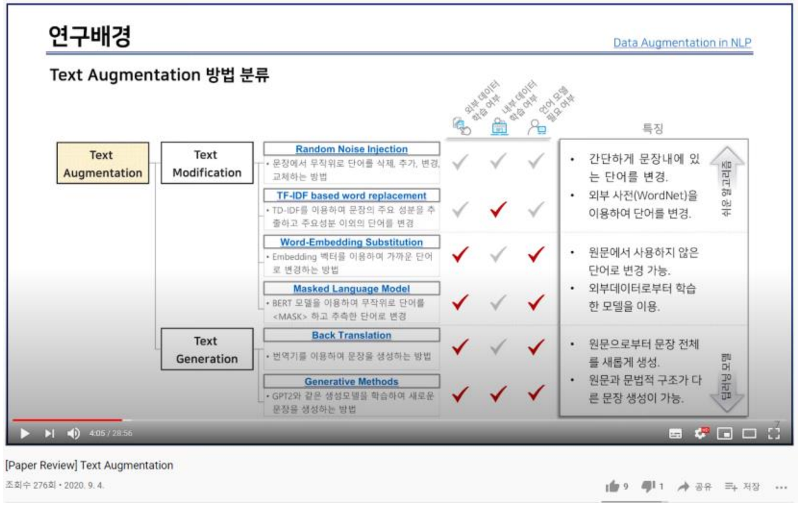
- Augmentation Methods
  - ✓ Word replacing with TF-IDF for Text Classification
    - Replaces uninformative words with low TF-IDF scores
    - While keeping those with high TF-IDF values

This virus has spread worldwide

↓

A virus has spread worldwide

- For more text augmentation methods, refer to Junghee Kim's presentation at DSBA seminar



https://youtu.be/UVtMqh3agQY

고려대학교 KOREA UNIVERSITY

DSBA Data Science & Business Analytics

Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

- Performance

Image Classification

Method	**Model**	**# Param**	CIFAR-10 (4k)	SVHN (1k)
Π-Model [32]	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher [58]	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin [41]	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG [35]	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
ICT [60]	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Pseudo-Label [33]	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT [25]	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
ICT [60]	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
MixMatch [3]	WRN-28-2	1.5M	6.24 ± 0.06	2.89 ± 0.06
Mean Teacher [58]	Shake-Shake	26M	6.28 ± 0.15	-
Fast-SWA [1]	Shake-Shake	26M	5.0	-
MixMatch [3]	WRN	26M	4.95 ± 0.08	-
UDA (RandAugment)	WRN-28-2	1.5M	4.32 ± 0.08	$\mathbf{2.23 \pm 0.07}$
UDA (RandAugment)	Shake-Shake	26M	3.7	-
UDA (RandAugment)	PyramidNet	26M	**2.7**	-

Text Classification

Fully supervised baseline

Datasets (# Sup examples)	IMDb (25k)	Yelp-2 (560k)	Yelp-5 (650k)	Amazon-2 (3.6m)	Amazon-5 (3m)	DBpedia (560k)
Pre-BERT SOTA	*4.32*	2.16	29.98	3.32	34.81	0.70
$\text{BERT}_{\text{LARGE}}$	4.51	*1.89*	*29.32*	*2.63*	*34.17*	*0.64*

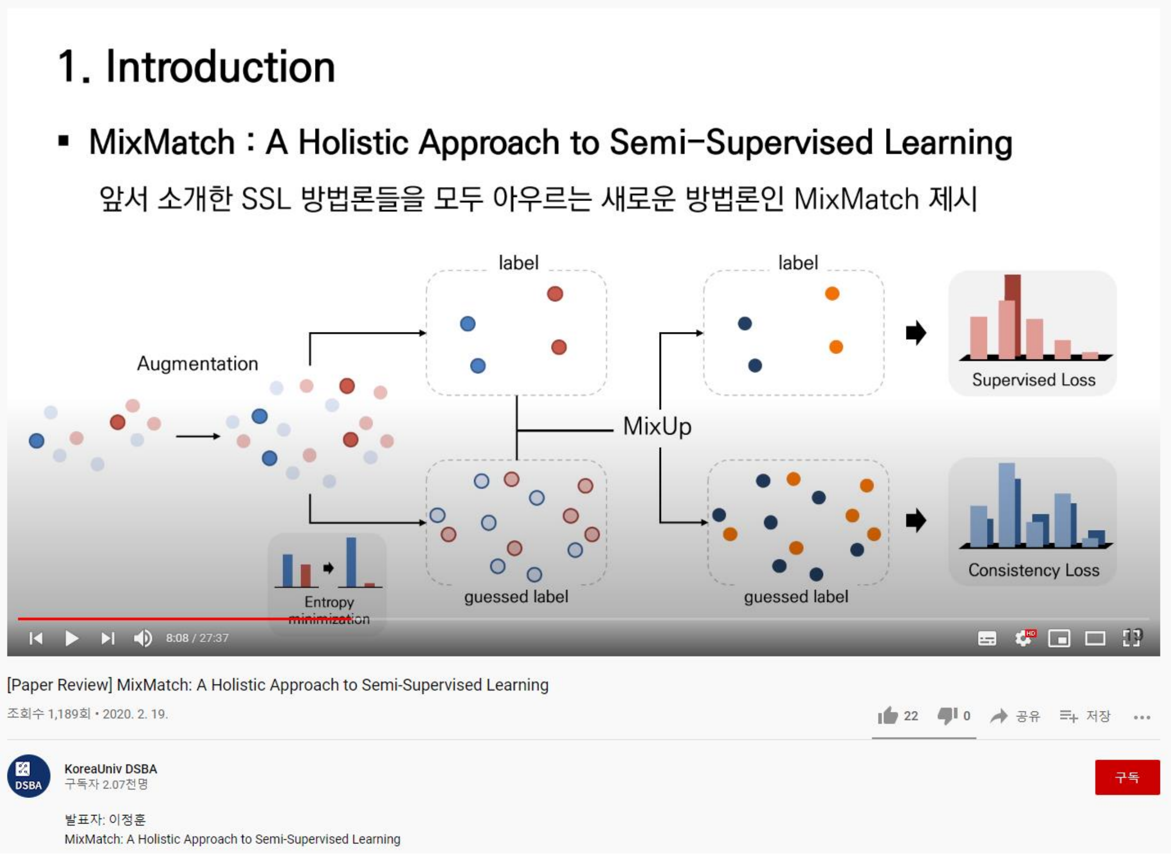
Semi-supervised setting

Initialization	**UDA**	IMDb (20)	Yelp-2 (20)	Yelp-5 (2.5k)	Amazon-2 (20)	Amazon-5 (2.5k)	DBpedia (140)
Random	✗	43.27	40.25	50.80	45.39	55.70	41.14
	✓	25.23	8.33	41.35	16.16	44.19	7.24
$\text{BERT}_{\text{BASE}}$	✗	18.40	13.60	41.00	26.75	44.09	2.58
	✓	5.45	2.61	33.80	3.96	38.40	1.33
$\text{BERT}_{\text{LARGE}}$	✗	11.72	10.55	38.90	15.54	42.30	1.68
	✓	4.78	2.50	33.54	3.93	37.80	1.09
$\text{BERT}_{\text{FINETUNE}}$	✗	6.50	2.94	32.39	12.17	37.32	-
	✓	**4.20**	**2.05**	**32.08**	**3.50**	**37.12**	-

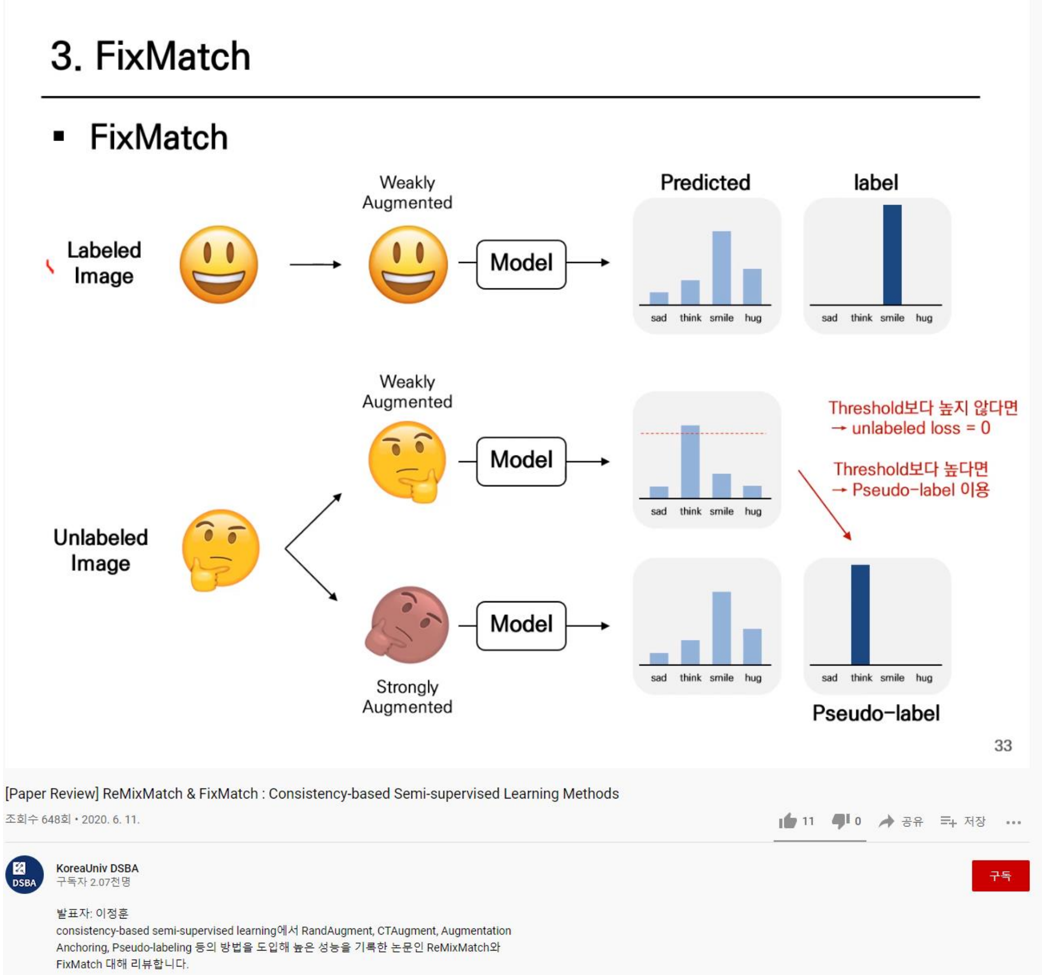
Unsupervised Data Augmentation for Consistency Training

Xie et al. (2019)

- For MixMatch, Remixmatch, and FixMatch, refer to Junghoon Lee's presentations at DSBA seminar



https://youtu.be/nSJP7bn2D1U



https://youtu.be/mXiPbkyGJ9g


Q & A

References

Research Papers

- Ben-David, S., Lu, T., & Pál, D. (2008, July). Does Unlabeled Data Provably Help? Worst-case Analysis of the Sample Complexity of Semi-Supervised Learning. In *COLT* (pp. 33-44).
- Bennett, K., & Demiriz, A. (1999). Semi-supervised support vector machines. *Advances in Neural Information processing systems*, 368-374.
- Blum, A., & Mitchell, T. (1998, July). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory* (pp. 92-100). ACM.
- Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020). Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (pp. 702-703).
- Fox-Roberts, P., & Rosten, E. (2014). Unbiased generative semi-supervised learning. *The Journal of Machine Learning Research*, *15*(1), 367-443.
- Kim, D., Seo, D., Cho, S., & Kang, P. (2017+). Multi-co-training for document classification using various document representations: TF-IDF, LDA, and Doc2Vec. under review.
- Kingma, D. P., Mohamed, S., Rezende, D. J., & Welling, M. (2014). Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems* (pp. 3581-3589).
- Singh, A., Nowak, R., & Zhu, X. (2009). Unlabeled data: Now it helps, now it doesn't. In *Advances in neural information processing systems* (pp. 1513-1520).
- Yu, S., Krishnapuram, B., Rosales, R., & Rao, R. B. (2011). Bayesian co-training. *The Journal of Machine Learning Research*, *12*, 2649-2680
- Zhou, Z. H., & Li, M. (2005, July). Semi-Supervised Regression with Co-Training. In IJCAI (Vol. 5, pp. 908-913).

References

Other materials

- Figures in the first page: 하상욱 단편시집 – 서울 시
- Zhu, X. (2007). Semi-Supervised Learning Tutorial. International Conference on Machine Learning (ICML 2007).
- Choi, S. (2015). Deep Learning: A Quick Overview. Deep Learning Workshop. KIISE.
- Zien, A. (2008). Semi-Supervised Learning. Summer School on Neural Networks.
- Zhu, X. (2009). Tutorial on Semi-Supervised Learning. Theory and Practice of Computational Learning.