d-SNE: Domain Adaptation using Stochastic Neighborhood Embedding

Xiang Xu[†,*], Xiong Zhou[‡,*], Ragav Venkatesan[‡], Gurumurthy Swaminathan[‡], Orchid Majumder[‡]
[†]Computational Biomedicine Lab, University of Houston, Houston, USA
[‡] AWS AI, Seattle, USA
[†]xxu18@central.uh.edu, [‡]{xiongzho,ragavven,gurumurs,orchid}@amazon.com

CVPR, 2019

Presented by : Kangyeol Kim
DAVIAN Lab, Korea University

Few-shot / Semi-supervised DA

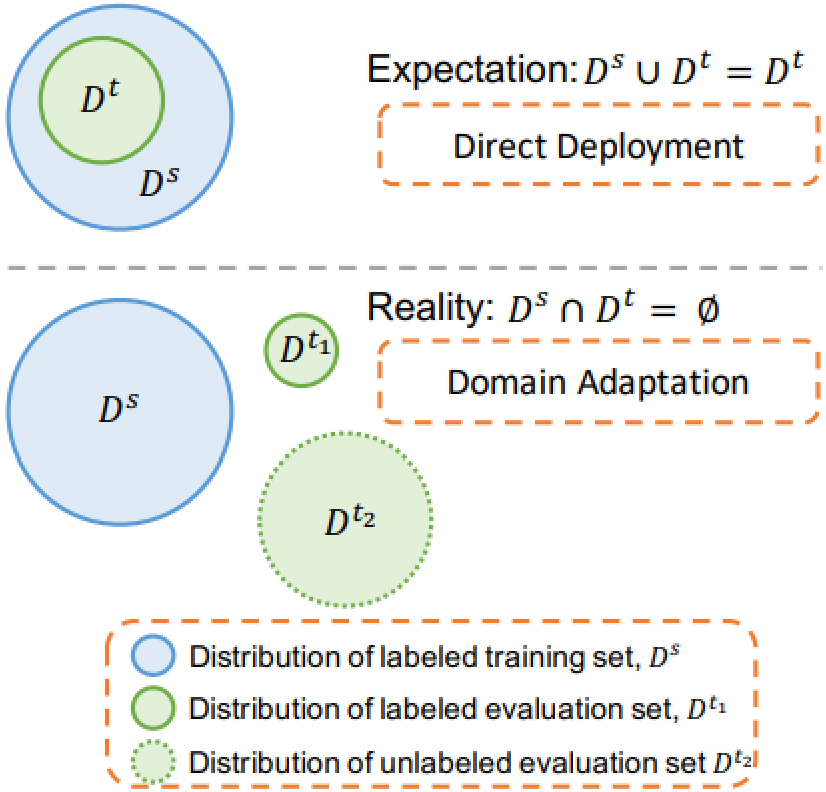
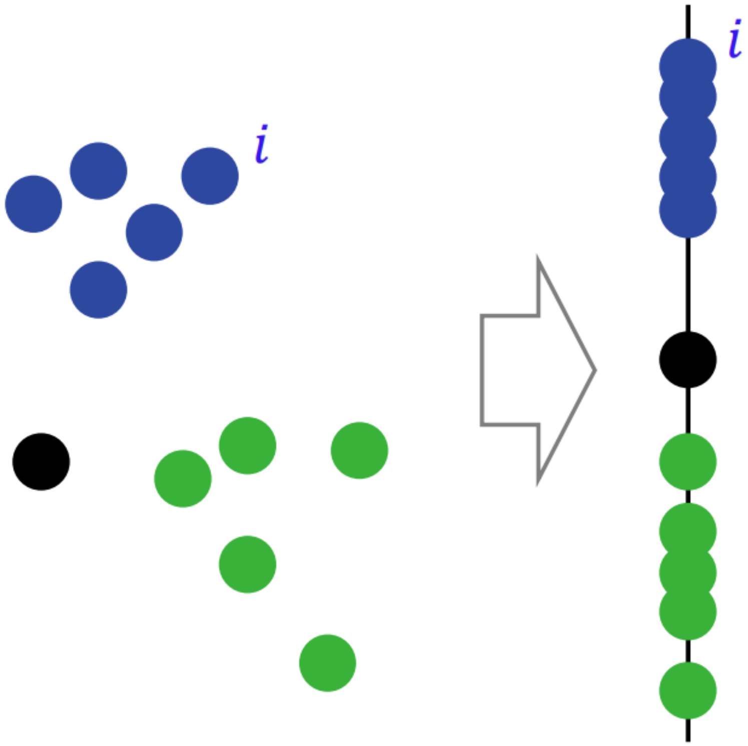
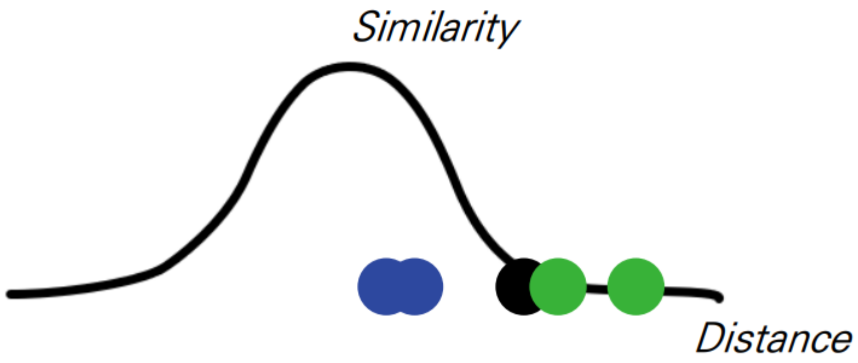


Figure 1: Domain adaptation in the true data space: Expectation vs. Reality.

Stochastic neighbor embedding

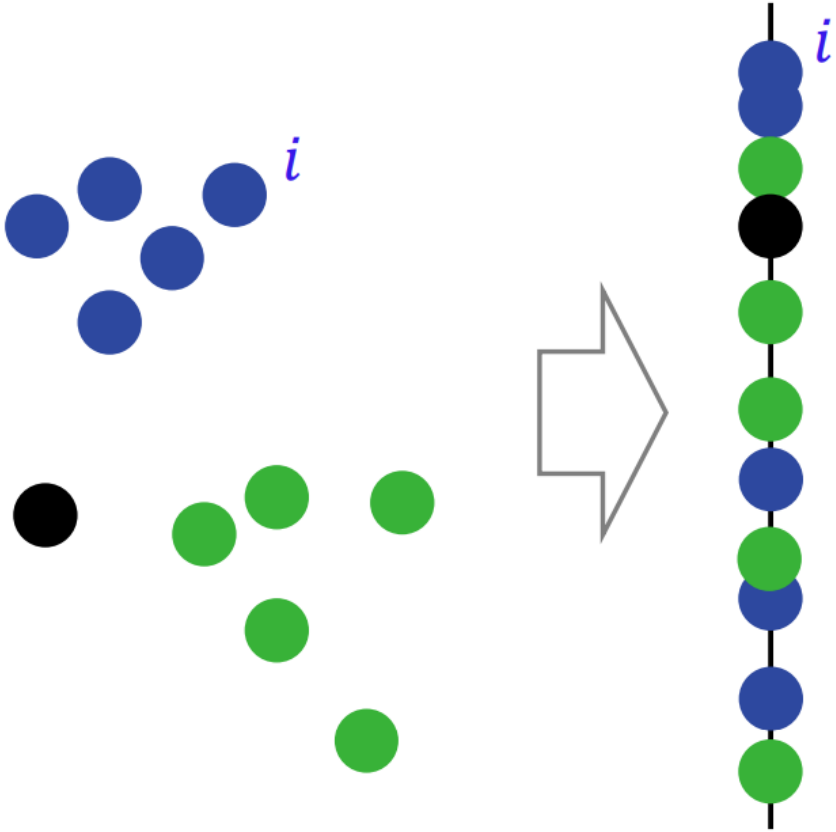


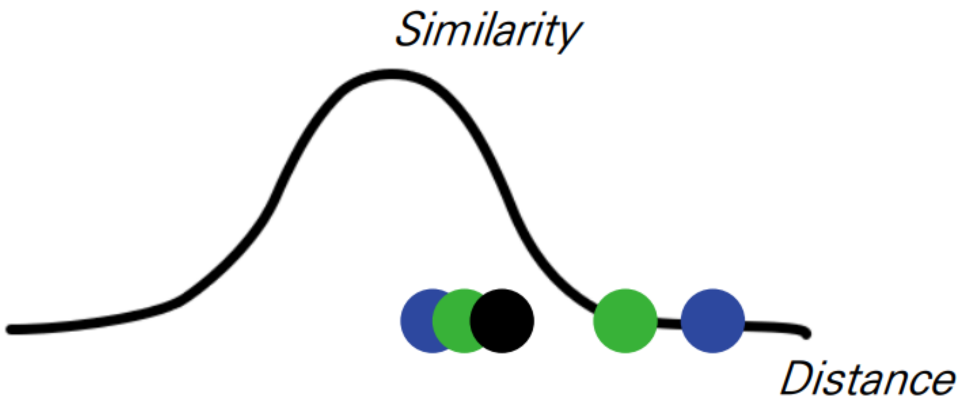


$$q_{j|i} = \frac{e^{-\|y_i - y_j\|^2}}{\sum_{k \neq i} e^{-\|y_i - y_k\|^2}}$$

Slide credit : Taeoh Kim

Stochastic neighbor embedding

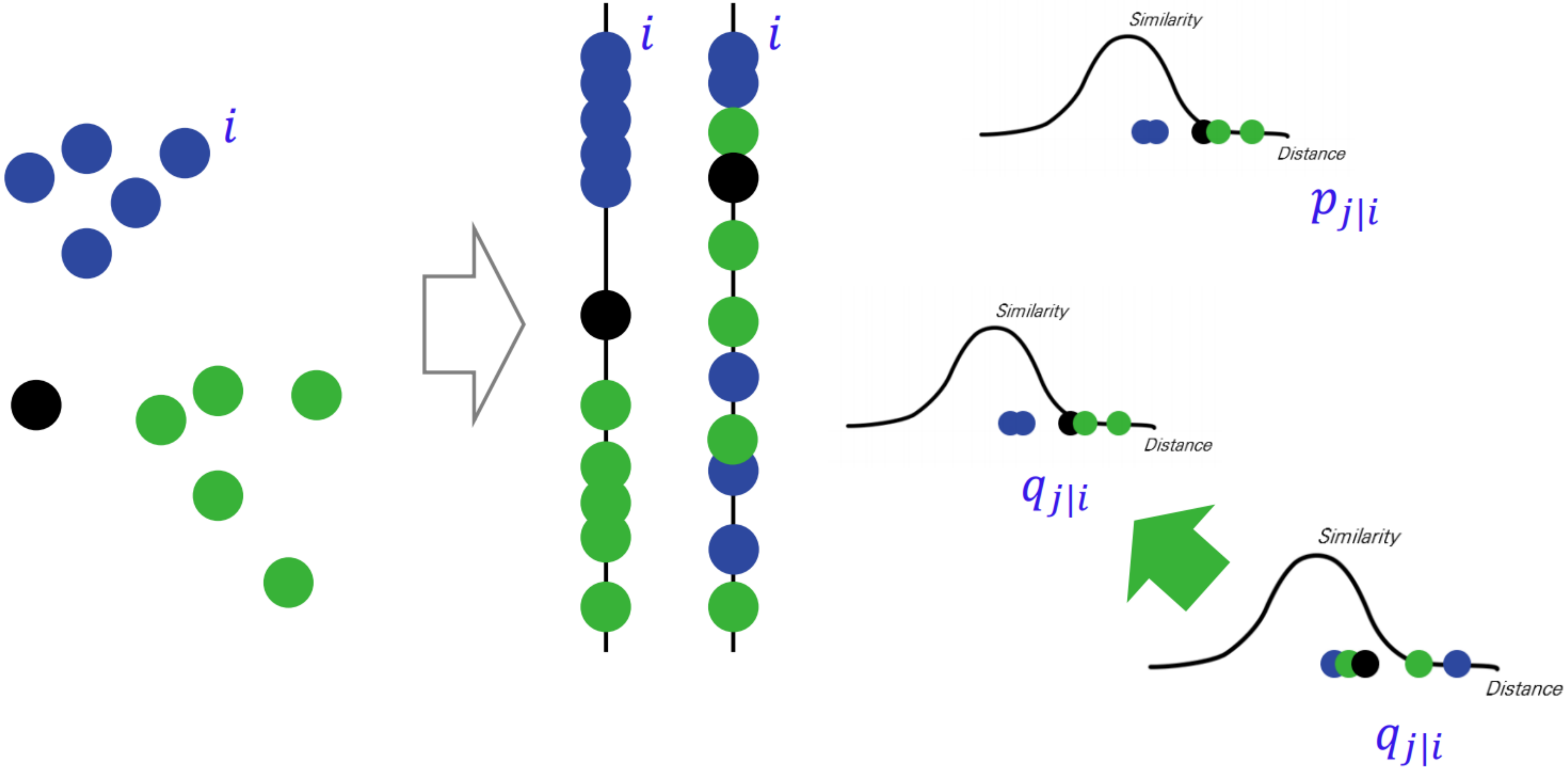




$$q_{j|i} = \frac{e^{-\|y_i - y_j\|^2}}{\sum_{k \neq i} e^{-\|y_i - y_k\|^2}}$$

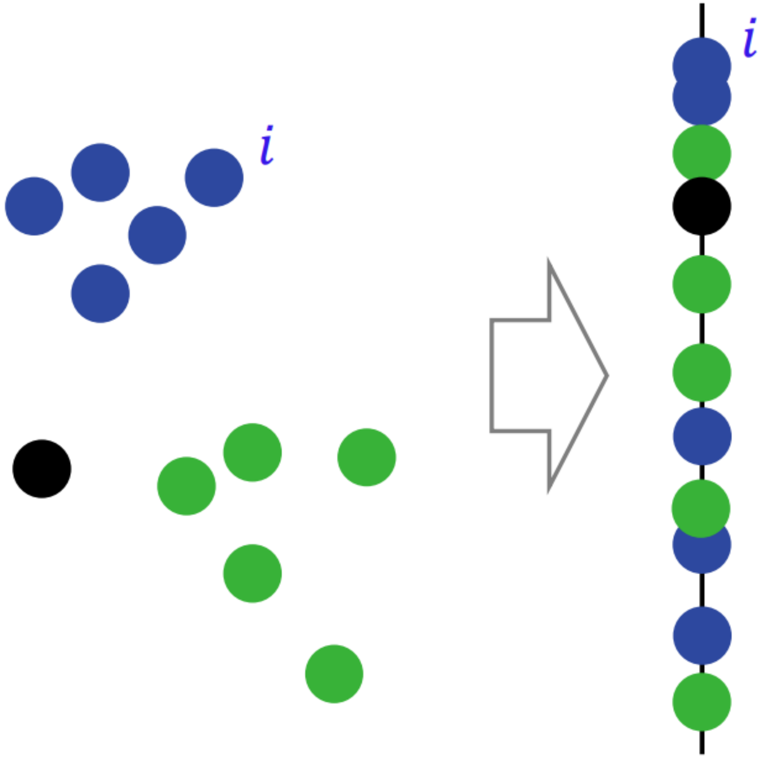
Slide credit : Taeoh Kim

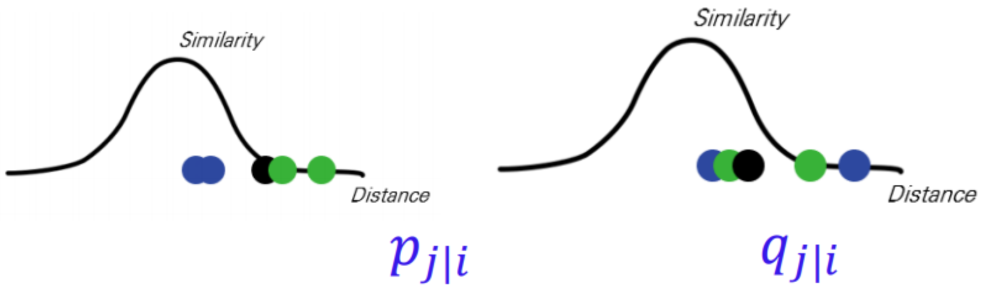
Stochastic neighbor embedding



Slide credit : Taeoh Kim

Stochastic neighbor embedding





$$C = KL(P||Q) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

for Every Data

$$\frac{\partial C}{\partial y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

Slide credit : Taeoh Kim

t-Stochastic neighbor embedding

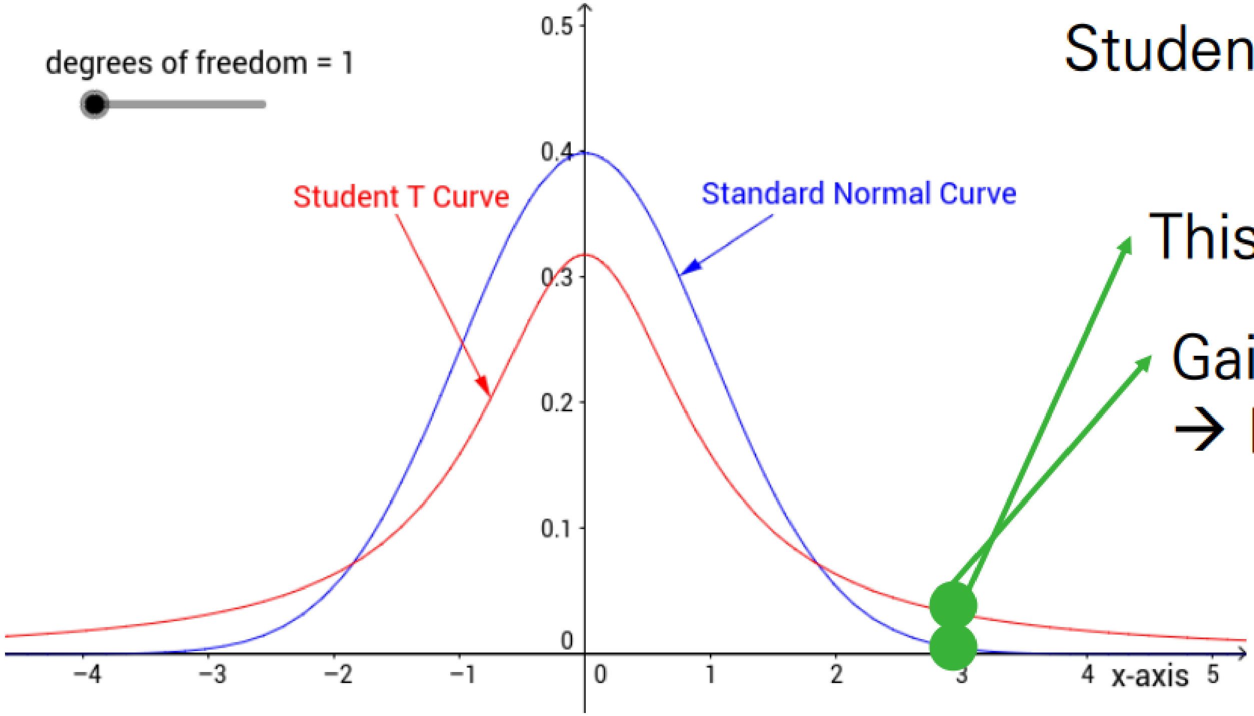
	SNE	Symmetric SNE	t-SNE
Prob. In High-D	$p_{j\|i} = \dfrac{e^{-\frac{\\|x_i-x_j\\|^2}{2\sigma_i^2}}}{\sum_{k\neq i} e^{-\frac{\\|x_i-x_k\\|^2}{2\sigma_i^2}}}$	$p_{ij} = \dfrac{e^{-\frac{\\|x_i-x_j\\|^2}{2\sigma^2}}}{\sum_{k\neq l} e^{-\frac{\\|x_k-x_l\\|^2}{2\sigma^2}}}$	$p_{ij} = \dfrac{p_{j\|i} + p_{i\|j}}{2n}$
Prob. In Low-D	$q_{j\|i} = \dfrac{e^{-\\|y_i-y_j\\|^2}}{\sum_{k\neq i} e^{-\\|y_i-y_k\\|^2}}$	$q_{ij} = \dfrac{e^{-\\|y_i-y_j\\|^2}}{\sum_{k\neq l} e^{-\\|y_k-y_l\\|^2}}$	$q_{ij} = \dfrac{\left(1+\\|y_i-y_j\\|^2\right)^{-1}}{\sum_{k\neq l}(1+\\|y_k-y_l\\|^2)^{-1}}$
Cost Function	$C = \sum_i \sum_j p_{j\|i} \log \dfrac{p_{j\|i}}{q_{j\|i}}$	$C = \sum_i \sum_j p_{ij} \log \dfrac{p_{ij}}{q_{ij}}$	
Gradient of Cost Function	$2\sum_j (p_{j\|i} - q_{j\|i} + p_{i\|j} - q_{i\|j})(y_i - y_j)$	$4\sum_j (p_{ij} - q_{ij})(y_i - y_j)$	$4\sum_j (p_{ij} - q_{ij})(y_i - y_j)\left(1+\\|y_i-y_j\\|^2\right)^{-1}$

PDF	$\dfrac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

PDF	$\dfrac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$

Slide credit : Taeoh Kim

t-Stochastic neighbor embedding



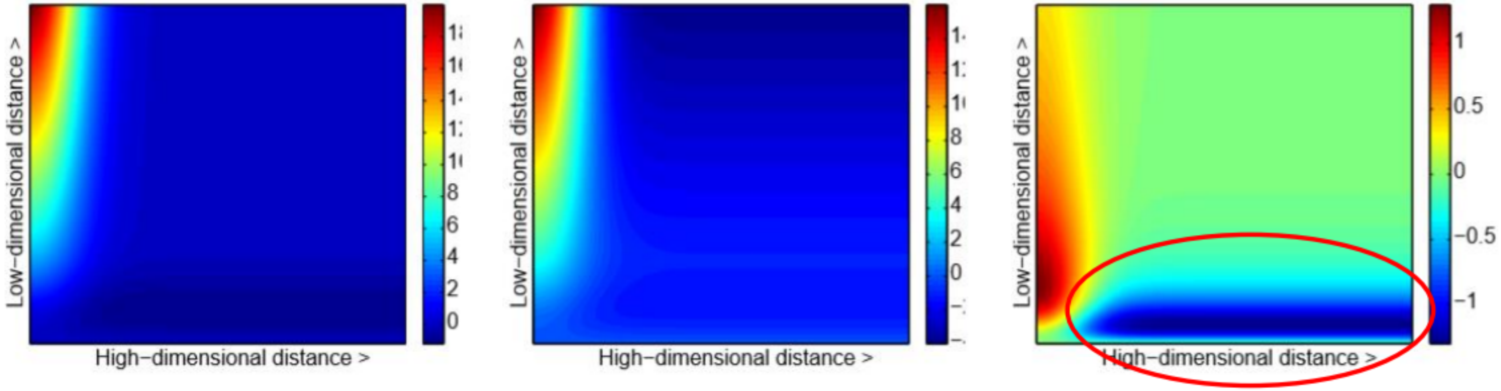
Student t−Distribution in Low−Dimension

This High−Dimension Data

Gains its Probability
→ More far away

Slide credit : Taeoh Kim

t-Stochastic neighbor embedding



(a) Gradient of SNE. (b) Gradient of UNI-SNE. (c) Gradient of t-SNE.

Strong Replusion

High−D	Low−D	p_{ij}	q_{ij}	$(p_{ij} - q_{ij})$	$(y_i - y_j)$	$\left(1 + \|y_i - y_j\|^2\right)^{-1}$	Gradient
Large	Large	1	1	0	Large	Small	0
Small	Small	0	0	0	Small	Large	0
Small	Large	0	1	−1	Large	Small	Attraction
Large	Small	1	0	1	Small	Large	Repulsion

Slide credit : Taeoh Kim

Motivation

To create this embedding space, we use a strategy that is very similar to the popular stochastic neighborhood embedding technique (SNE) [12]. To modifiy SNE for domain adaptation, we use a novel modified-Hausdorff distance metric in a $\min - \max$ formulation. d-SNE minimizes the distance between the samples from $\mathcal{D}^s$ and $\mathcal{D}^t$ so as to maximize the margin of inter-class distance for discrimination and minimize the intra-class distance from both domains to achieve domain-invariance. This discrimination is learnt as a max-margin nearest-neighbor form to make the network optimization easy. Our proposed idea is still learnable in an end-to-end fashion, therefore making it ideal for training neural networks.

Proposed method

- Point-to-point relationship between source domain and target domain
- Probability of being same class with point in source domain can be defined as:

$$p_{ij} = \frac{\exp(-d(x_i^s, x_j^t))}{\sum_{x \in \mathcal{D}^s} \exp(-d(x, x_j^t))}.$$

Proposed method

- Points-to-point relationship between source domain and target domain
- Probability of being specific class (1..k) can be defined as:

$$p_j = \frac{\sum_{x \in \mathcal{D}_k^s} \exp(-d(x, x_j^t))}{\sum_{x \in \mathcal{D}^s} \exp(-d(x, x_j^t))} = \sum_{i=0}^{N_k^s} p_{ij},$$

- It corresponds with multinomial distribution as softmax output

Proposed method

- The objective function to minimize is then,

$$\sum_{x_j \in \mathcal{D}^t} \frac{1}{p_j} = \sum_{x_j \in \mathcal{D}^t} \left(\frac{\sum_{x \in \mathcal{D}^s_{\not k}} \exp(-d(x, x_j))}{\sum_{x \in \mathcal{D}^s_k} \exp(-d(x, x_j))}, \text{ for } k = y_j \right). \tag{4}$$

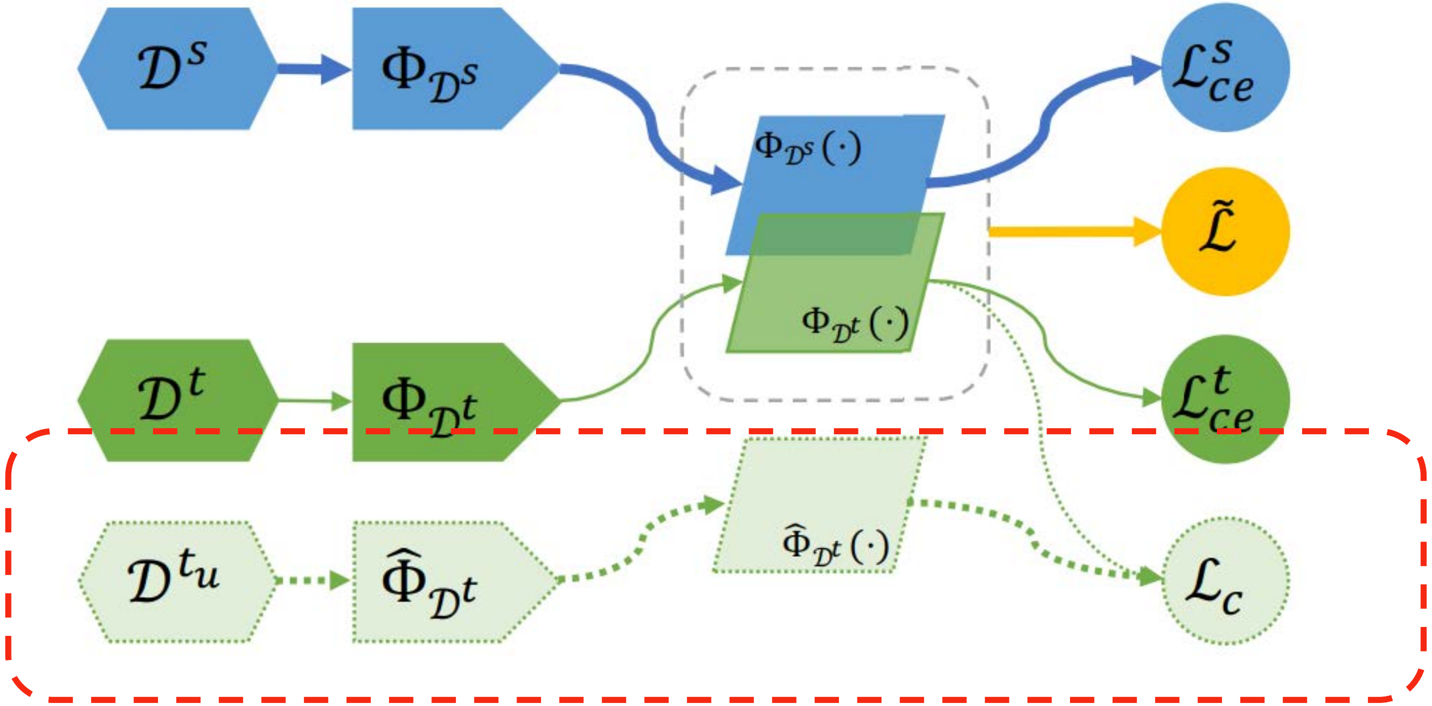
- Numerator – inter-class distances
- Denominator – intra-class distances

Proposed method

- The form of sum of exponentials leads to adverse effects in stochastic optimization due to scaling issue.
- Relax likelihood with the use of a modified-Hausdorffian distance.
- Only minimizing largest distance between the samples of the same class and maximize the smallest distance between the samples of different classes

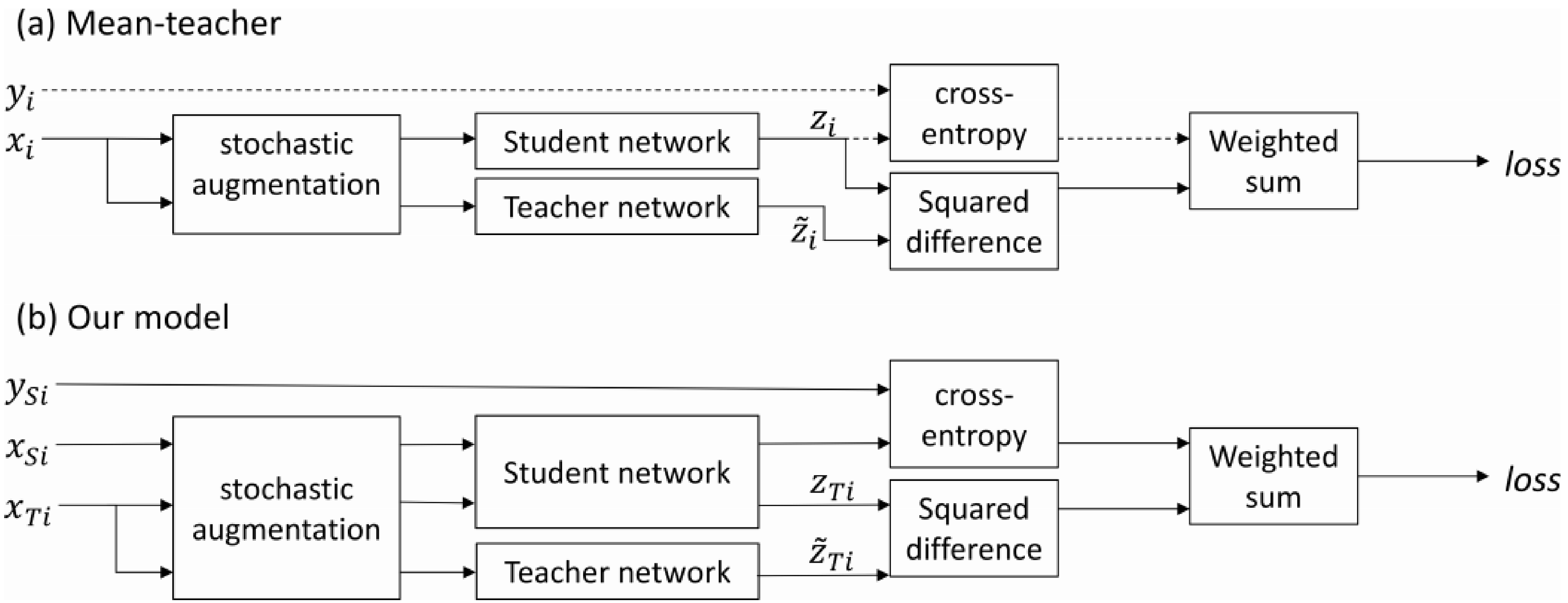
$$\tilde{\mathcal{L}} = \sup_{x \in \mathcal{D}_k^s} \{a | a \in d(x, x_j)\} - \inf_{x \in \mathcal{D}_{k\!\!\!/}^s} \{b | b \in d(x, x_j)\}, \quad \text{for } k = y_j .$$

Proposed method



$$\underset{w_s, w_d}{\operatorname{argmin}} \tilde{\mathcal{L}} + \alpha \mathcal{L}^s_{\text{ce}} + \beta \mathcal{L}^t_{\text{ce}}$$

Proposed method



French, G et al., "Self-ensembling for visual domain adaptation", ICLR, 2018

Experiment

Method	$\lvert\mathcal{D}_k^t\rvert,\ \forall k$	Setting	A → D	A → W	D → A	D → W	W → A	W → D	Avg.
DANN [7]		$\mathcal{U}$	-	73.00	-	96.40	-	99.20	-
DRCN [8]			67.10 ± 0.30	68.70 ± 0.30	56.00 ± 0.50	96.40 ± 0.30	54.09 ± 0.50	99.00 ± 0.2	73.60
kNN-Ad [24]			84.10	81.10	58.30	96.40	63.80	99.20	80.48
I2I [17]			71.10	75.30	50.10	96.50	52.10	99.60	74.12
G2A [23]			87.70 ± 0.50	89.50 ± 0.50	72.80 ± 0.30	97.90 ± 0.30	71.40 ± 0.40	99.8 ± 0.4	86.50
SDA [27]	3	$\mathcal{S}$	86.10 ± 1.20	82.70 ± 0.80	66.20 ± 0.30	95.70 ± 0.50	65.00 ± 0.5	97.60 ± 0.20	82.22
FADA [15]	3		88.20 ± 1.00	88.10 ± 1.20	68.10 ± 0.60	96.40 ± 0.80	71.10 ± 0.90	97.50 ± 0.90	84.90
CCSA [16]	0		61.20 ± 0.90	62.3 ± 0.80	58.5 ± 0.80	80.1 ± 0.60	51.6 ± 0.90	95.6 ± 0.70	68.20
CCSA [16]	3		89.00 ± 1.20	88.20 ± 1.00	71.80 ± 0.50	96.40 ± 0.80	72.10 ± 1.00	97.60 ± 0.40	85.80
d-SNE (VGG-16)	0	$\mathcal{S}$	62.40 ± 0.40	61.49 ± 0.75	48.92 ± 1.03	82.24 ± 1.42	47.52 ± 0.94	90.42 ± 1.00	65.49
	3		91.44 ± 0.23	90.13 ± 0.07	71.06 ± 0.18	97.10 ± 0.07	71.74 ± 0.42	97.46 ± 0.24	86.49
d-SNE (ResNet-101)	0	$\mathcal{S}$	80.41 ± 0.79	75.26 ± 1.32	67.39 ± 0.18	96.39 ± 0.41	65.55 ± 1.91	98.31 ± 1.87	80.55
	3		$\mathbf{94.65 \pm 0.38}$	$\mathbf{96.58 \pm 0.14}$	$\mathbf{75.51 \pm 0.44}$	$\mathbf{99.10 \pm 0.24}$	$\mathbf{74.20 \pm 0.24}$	$\mathbf{100.00 \pm 0.00}$	**90.01**

Proposed methods, 1D example

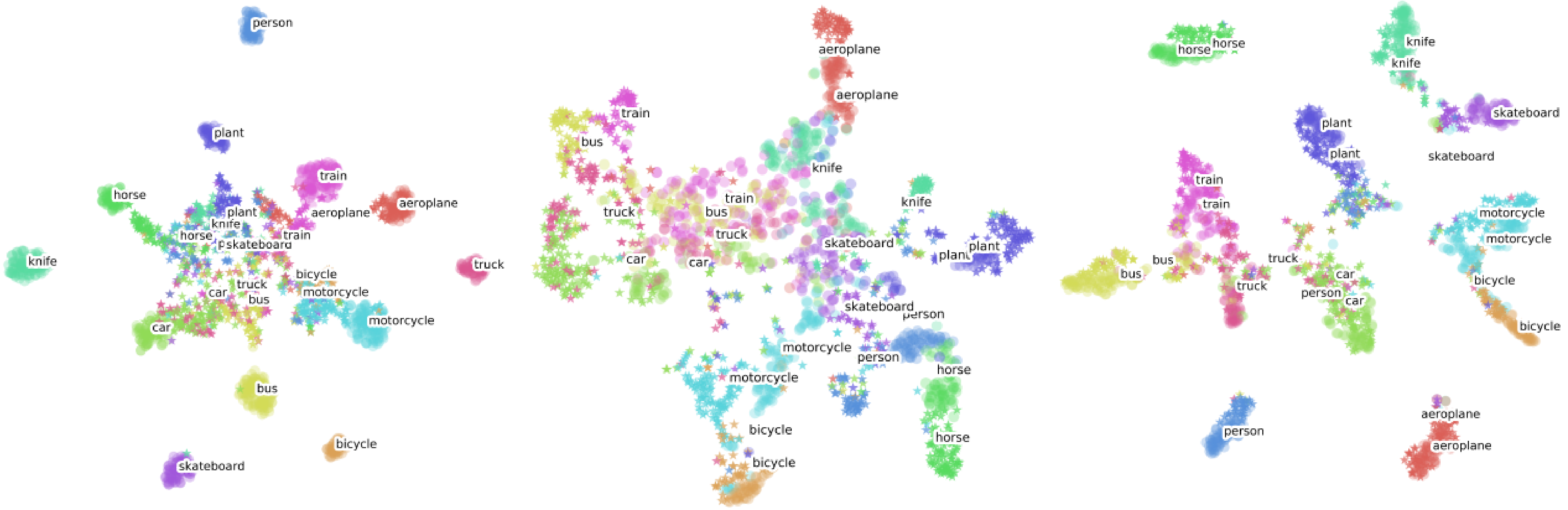


Figure 6: t-SNE visualization of d-SNE's latent-embedding space for the VisDA-C dataset. (a) Embeddings produced by the model trained with source images only. (b) Embeddings produced by the model trained with target images only and (c) The joint latent-embedding space of d-SNE. Different colors represent different classes. Embeddings from the source and target domains are indicated by circles and stars, respectively.