GCAN:
Graph Convolutional Adversarial Network for **Unsupervised Domain Adaptation**

(X. Ma, T. Zhang, and C. Xu. In CVPR 2019)

컴퓨터공학부 2020-39297 최은태

Unsupervised Domain Adaptation

General domain adaptation problem

Source data D_S → Target data D_T

Covariate shift

Categorization by label accessibility

Supervised Domain Adaptation (SDA): Labeled D_S → Labeled D_T

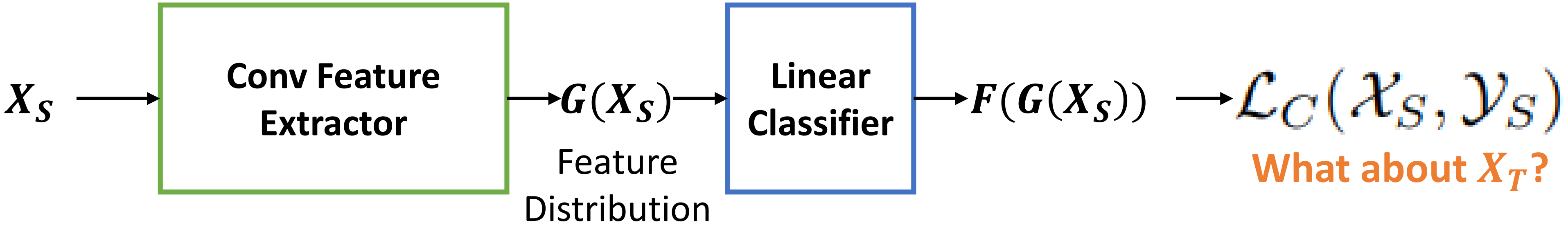
- **Jointly train on D_S & D_T with full-supervision**

Unsupervised Domain Adaptation (UDA): Labeled D_S → Unlabeled D_T

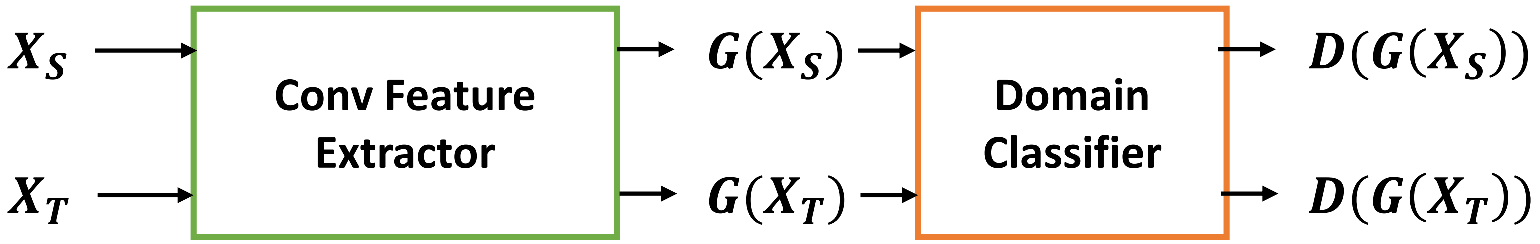
- **Should align the feature distributions of D_T to source domain**

Adversarial Training-Based Approach

Consider a CNN with **a feature extractor G** & **a linear classifier F**



Adopt the idea of GAN: Make the features domain-invariant to fool **a domain classifier D**



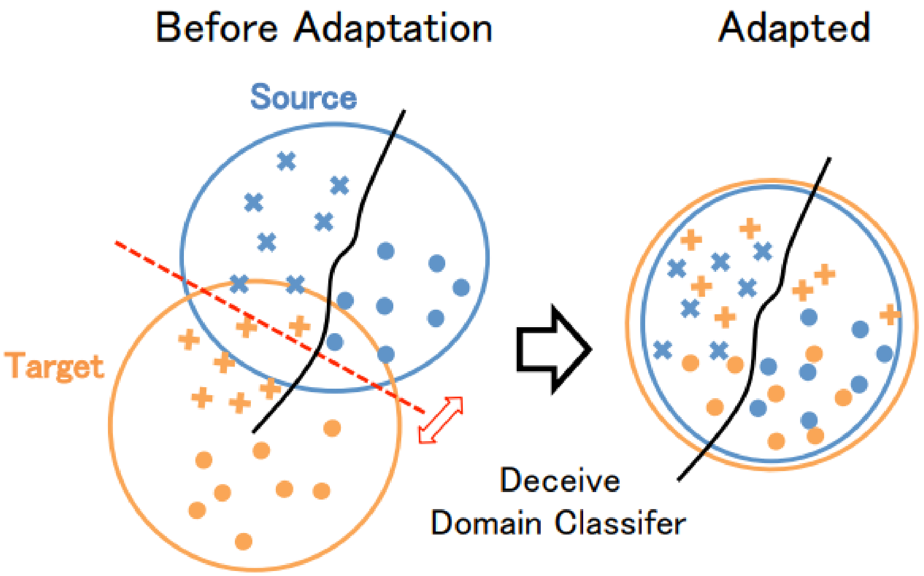
$$d(\mathcal{X}_S, \mathcal{X}_T) = \mathbb{E}_{x\sim D_S}[\log(1 - \mathrm{D} \circ \mathrm{G}(\mathrm{x}))]$$
$$\mathbb{E}_{x\sim D_T}[\log(\mathrm{D} \circ \mathrm{G}(\mathrm{x}))]$$

MSTN: The predecessor of GCAN

Adversarial training alone is not enough

- It can map target features **near source features**
- **But class discriminability is not guaranteed**

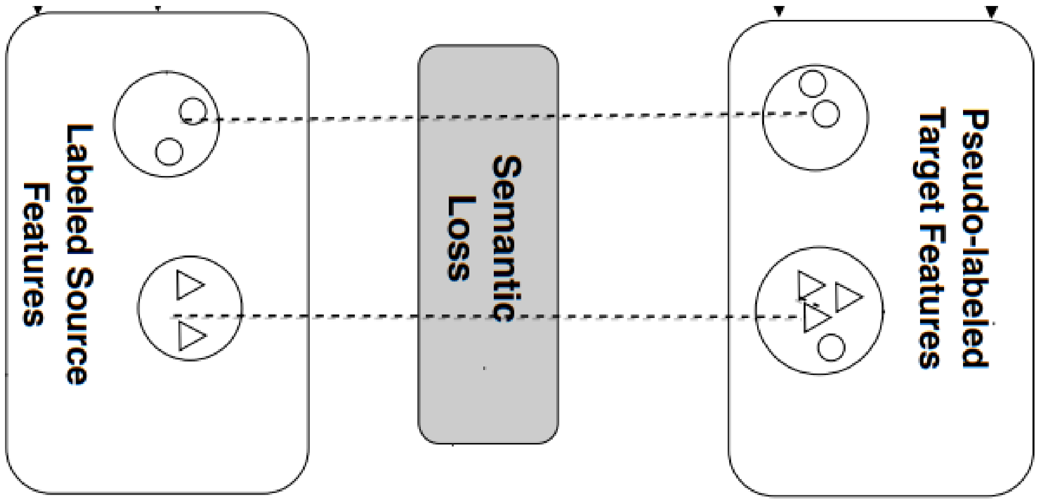
➡ **Per-class alignment loss**



$$\mathcal{L}^{UDA}_{SM}(\mathcal{X}_S,\mathcal{Y}_S,\mathcal{X}_T)=\underbrace{\sum_{k=1}^{K}\Phi(C^k_S,C^k_T)}_{\mathcal{L}_{SM}(\mathcal{X}_S,\mathcal{Y}_S,\mathcal{X}_T)},$$

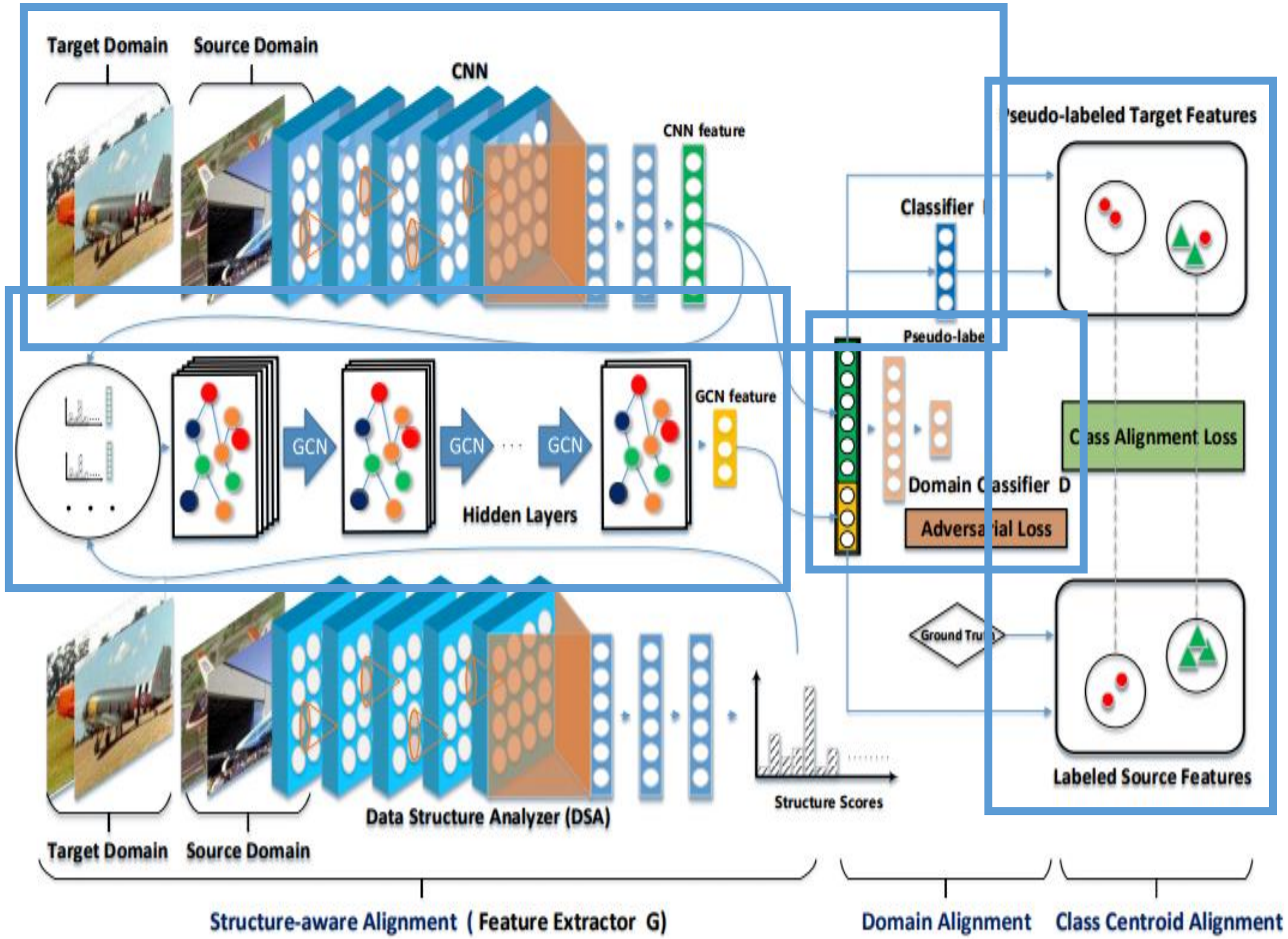
Φ: distance function

C^k: the centroid of features from class k



MSTN: http://proceedings.mlr.press/v80/xie18c/xie18c.pdf

GCAN Architecture & Loss Functions



1. $\mathcal{L}_C = \mathbb{E}_{(x,y)\sim D_S}[cross\_entropy(F(G(x)), y)]$
2. $\mathcal{L}_{DA} = \mathbb{E}_{x\in D_S}[\log(1 - D(G(x))] + \mathbb{E}_{x\in D_T}[\log(D(G(x))]$
3. $\mathcal{L}_{CA} = \sum_{k=1}^{K} \phi(C_S^k, C_T^k)$
4. $\mathcal{L}_{SA} = \max(\left\| X_{SC_a} - X_{SC_p} \right\|^2 - \left\| X_{SC_a} - X_{SC_n} \right\|^2 + m, 0)$

Linear combination to get the total loss

$$\mathcal{L}_{total} = \mathcal{L}_C + \lambda\mathcal{L}_{DA} + \gamma\mathcal{L}_{CA} + \eta\mathcal{L}_{SA}$$

Adopting GCN

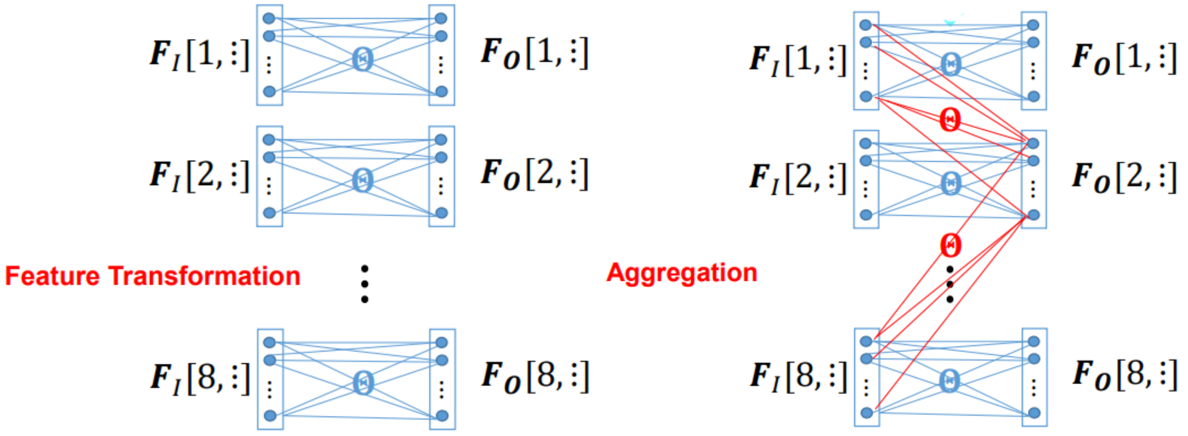
$\mathcal{L}_{SA} = \max(\left\|X_{SC_a} - X_{SC_p}\right\|^2 - \left\|X_{SC_a} - X_{SC_n}\right\|^2 + m, 0)$ => **TripletLoss** on GCN features

$$X = G(X_{batch})$$

$X_{SC} = \widehat{D}^{-\frac{1}{2}}\hat{A}\widehat{D}^{-\frac{1}{2}}X^{T}W$ (<=Simplified ChebNet)

$$\hat{A} = A + I = XX^{T}$$

$$\widehat{D}_{ii} = \sum_{j} \hat{A}_{ij}$$



CNN vs GCN

- As addressed in the lecture, GCN features are aware of **instance relation**
- Then, $\mathcal{L}_{SA}$ **aligns the data structure in feature space**

TripletLoss: https://arxiv.org/pdf/1412.6622.pdf

Experimental Settings

Base CNN: AlexNet

Dimension of features: 256

Domain classifier: $256\ (feat) \rightarrow 1{,}024 \rightarrow 1{,}024 \rightarrow 1$

GCN: a single-layer Simplified ChebNet ($256\ (feat) \rightarrow 150$)

Datasets used

Name	# of Classes	# of Domains	# of Images
ImageCLEF-DA	12	3	~1.8k
Office-31	31	3	~4k
Office Home	65	4	~30k

Experimental Results

Table 2. Classification accuracy (%) on ImageCLEF-DA dataset.

Method	$I \rightarrow P$	$P \rightarrow I$	$I \rightarrow C$	$C \rightarrow I$	$C \rightarrow P$	$P \rightarrow C$	Avg
AlexNet	66.2±0.2	70.0±0.2	84.3±0.2	71.3±0.4	59.3±0.5	84.5±0.3	73.9
RTN [52]	67.4±0.3	81.3±0.3	89.5±0.4	78.0±0.2	62.0±0.2	89.1±0.1	77.9
RevGrad [22]	66.5±0.5	81.8±0.4	89.0±0.5	79.8±0.5	63.5±0.4	88.7±0.4	78.2
JAN [53]	67.2±0.5	82.8±0.4	91.3±0.5	80.0±0.5	63.5±0.4	91.0±0.4	79.3
MSTN [77]	67.3±0.3	82.8±0.2	91.5±0.1	81.7±0.3	65.3±0.2	91.2±0.2	80.0
GCAN	**68.2±0.5**	**84.1±0.2**	**92.2±0.1**	**82.5±0.1**	**67.2±0.2**	**91.3±0.1**	**80.9**

Table 1. Classification accuracy (%) on the Office-31 dataset.

Method	$A \rightarrow W$	$D \rightarrow W$	$W \rightarrow D$	$A \rightarrow D$	$D \rightarrow A$	$W \rightarrow A$	Avg
AlexNet	61.6±0.5	95.4±0.3	99.0±0.2	63.8±0.5	51.1±0.6	49.8±0.4	70.1
DDC [74]	61.8±0.4	95.0±0.5	98.5±0.4	64.4±0.3	52.1±0.6	52.2±0.4	70.6
DRCN [27]	68.7±0.3	96.4±0.3	99.0±0.2	66.8±0.5	56.0±0.5	54.9±0.5	73.6
RevGrad [22]	73.0±0.5	96.4±0.3	99.2±0.3	72.3±0.3	53.4±0.4	51.2±0.5	74.3
RTN [52]	73.3±0.3	96.8±0.2	99.6±0.1	71.0±0.2	50.5±0.3	51.0±0.1	73.7
JAN [53]	74.9±0.3	96.6±0.2	99.5±0.2	71.8±0.2	58.3±0.3	55.0±0.4	76.0
AutoDIAL [50]	75.5	96.6	99.5	73.6	58.1	59.4	77.1
MSTN [77]	80.5±0.4	96.9±0.1	**99.9±0.1**	74.5±0.4	62.5±0.4	60.0±0.6	79.1
GCAN	**82.7±0.1**	**97.1±0.1**	99.8±0.1	**76.4±0.5**	**64.9±0.1**	**62.6±0.3**	**80.6**

Table 3. Recognition accuracies (%) for cross-domain experiments on the Office+Home dataset.

Source	Ar	Ar	Ar	Cl	Cl	Cl	Pr	Pr	Pr	Rw	Rw	Rw	Avg
Target	Cl	Pr	Rw	Ar	Pr	Rw	Ar	Cl	Rw	Ar	Cl	Pr	
GFK [29]	21.60	31.72	38.83	21.63	34.94	34.20	24.52	25.73	42.92	32.88	28.96	50.89	32.40
JDA [50]	25.34	35.98	42.94	24.52	40.19	40.90	25.96	32.72	49.25	35.10	35.35	55.35	36.97
CCSL [57]	23.51	34.12	40.02	22.54	35.69	36.04	24.84	27.09	46.36	34.61	31.75	52.89	34.12
LSC [38]	31.81	39.42	50.25	35.46	51.19	51.43	30.46	39.54	59.74	43.98	42.88	62.25	44.87
RTML [17]	27.57	36.20	46.09	29.49	44.69	44.66	28.21	36.12	52.99	38.54	40.62	57.80	40.25
JGSA [86]	28.81	37.57	48.92	31.67	46.30	46.76	28.72	35.90	54.47	40.61	40.83	59.16	41.64
PUnDA [28]	29.99	37.76	50.17	33.90	48.91	48.71	30.31	38.69	56.91	42.25	44.51	61.05	43.60
DAN [49]	30.66	42.17	54.13	32.83	47.59	49.78	29.07	34.05	56.70	43.58	38.25	62.73	43.46
DHN [75]	31.64	40.75	51.73	34.69	51.93	52.79	29.91	39.63	60.71	44.99	45.13	62.54	45.54
WDAN [79]	32.26	43.16	54.98	34.28	49.92	50.26	30.82	38.27	56.87	44.32	39.35	63.34	44.82
GAKT [18]	34.49	43.63	55.28	36.14	52.74	53.16	31.59	40.55	61.43	45.64	44.58	64.92	47.01
MSTN [77]	34.87	46.20	56.77	36.63	54.97	55.41	33.27	41.66	60.62	46.94	45.90	68.25	48.46
GCAN	**36.43**	**47.25**	**61.08**	**37.90**	**58.25**	**57.00**	**35.77**	**42.66**	**64.47**	**50.08**	**49.12**	**72.53**	**51.05**

Implementation Details

Based on MSTN implementation https://github.com/wgchang/DSBN

Since the pre-trained AlexNet model as modified in GCAN paper was not available in PyTorch, all experiments are done with pre-trained ResNet18 (largest model that can run in a single TitanX Pascal GPU)

Using SGD as in GCAN paper w/ lr=1e-2 always leads to divergence => Adam w/ lr=1e-5 is used

When the concatenated features (CNN + GCN) are fed to both classifier F & domain classifier D, the cross-entropy loss and the centroid alignment loss are too large initially (> 30,000)
⇒ **leads to degenerate solution even after loss convergence**
⇒ **therefore, the concatenated features are fed to D, and the CNN features are fed to F**

Many details (GCN initialization, warmup learning rate, data augmentation, etc.) were missing in GCAN paper
⇒ **followed the hyperparameter settings of the MSTN repository**

Reproduced Results

Office-31, ResNet-18, classification accuracy (%)

Method	A->D	A->W	D->A	D->W	W->A	W->D	Avg
MSTN	81.93	81.76	65.53	97.99	59.89	99.80	81.15
GCAN	74.30	74.09	56.48	97.48	58.54	100.0	76.82

A->D, A->W, D->A experiments are still running (only ~21% of total iteration)
=> will be updated in the **results.docx** file in the link after the training is finished

How to Run the Codes

1. The codes can be downloaded here
2. Download Office-31 dataset from here
3. PyTorch >= 1.3, python=3.6, h5py, opencv (refer to https://github.com/wgchang/DSBN)
4. Modify '**OFFICE_DIR**' variable in dataset/factory.py to **your own office31 top folder**
5. Run the following command

python trainval_multi.py --model-name resnet18 --exp-setting office --in-features 256 --sa-loss --sm-loss --adv-loss --source-datasets webcam --target-datasets amazon --batch-size 40 --save-dir output/office_wa --print-console --gpu 0

This command runs GCAN on W(ebcam)->A(mazon) scenario on gpu0
Choose --source-datasets & --target-datasets among ['amazon', 'dslr', 'webcam']
About 9GB of VRAM is required

Currently, only ResNet18 on Office31 is supported

Thank you!