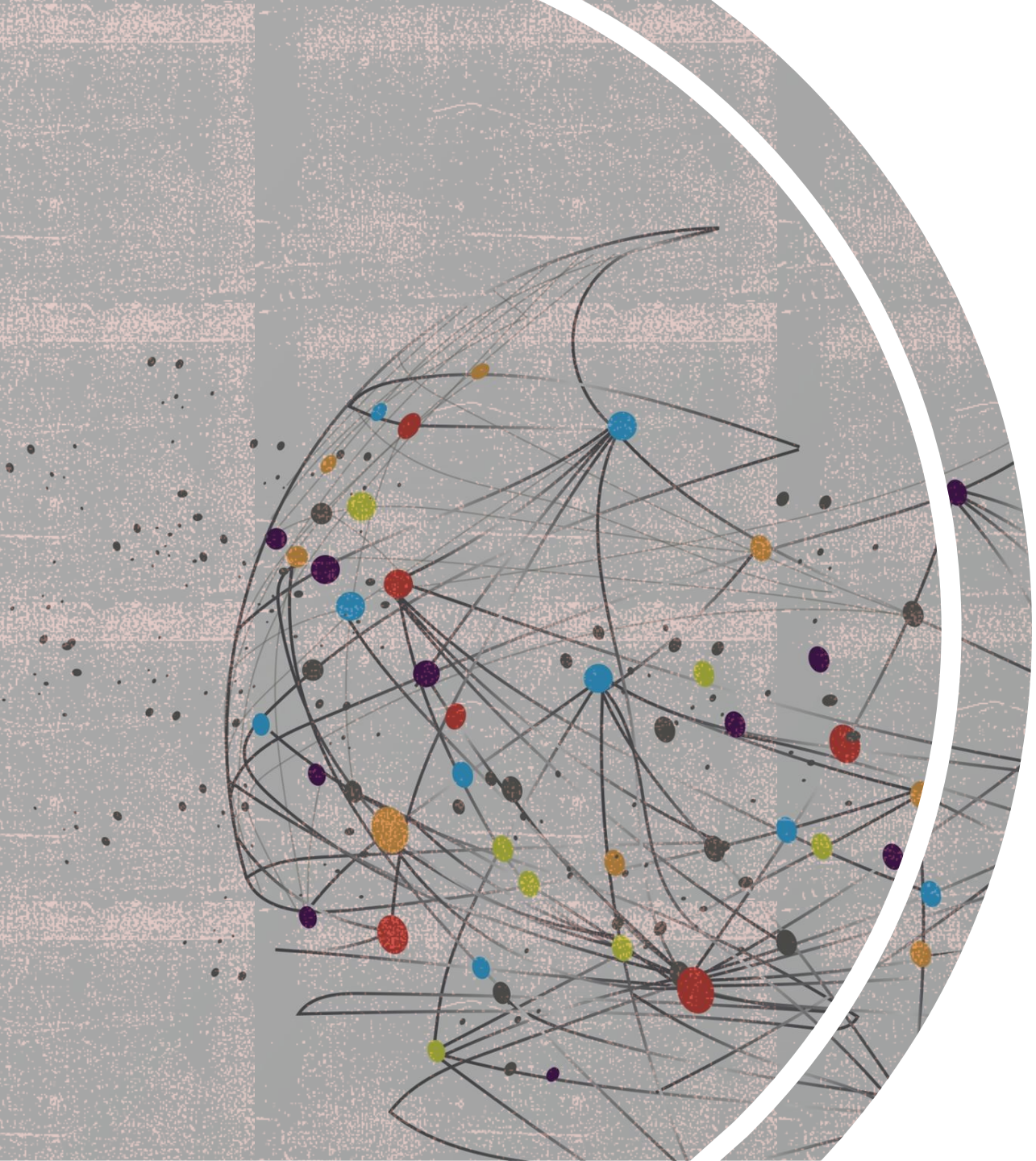


서울과학기술대학교
SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

Big Data Management and Application Laboratory

AN INCREMENTAL DEEP CONVOLUTIONAL COMPUTATION MODEL FOR FEATURE LEARNING ON INDUSTRIAL BIG DATA

Presentor: Minseon kim

BACKGROUND

Advanced computational techniques	vs	Robust model with lower training cost

- 데이터는 생성되는 속도가 빠르고, distribution이 넓다. 따라서 이기종 데이터의 feature engineering을 위해서는 robust함과 동시에 incremental한 learning을 가능하게 하는 모델이 필수적임.

A. *Parameter-incremental algorithm*: 훈련된 모델의 가중치와 bias를 수정하는 몇 가지 새로운 제약 규칙을 설계하여 new input을 반영합니다. 또한 new input은 sequential 하게 모델에 공급됩니다. 전체 데이터 세트를 메모리에 유지할 필요가 없다.

B. *Structure incremental algorithmn*: 새로운 입력의 특징을 포착하는 새로운 multilayer perceptron이 네트워크 구조를 업데이트할 때 pre-trained classifier의 앙상블에 추가된다.

INTRODUCTION

- Parameter-incremental learning
  1. 초기 parameter조정을 위해서 새로운 loss function 도입
  2. 새로운 Dropout strategy 도입; FCN에서의 뉴런들은 3개의 하위집합으로 그룹화되어 idle한 뉴런의 새로운 패턴에 대한 학습을 위해서 서로 다른 switch probability를 가진다.
  3. Fine tuning 학습을 통해 기존의 knowledge에 현재 knowledge 병합

- Structure-incremental learning
  1. Tensor convolutional, pooling, and FCN layers가 과거 정보를 transfer하도록 설계
  2. 0의 weight를 가지는 Virtual neurons을 도입
  3. Tensor space로의 Dropout 확장

DCCM

- Deep Convolutional Computation Model(DCCM)
  : 이기종 데이터의 feature engineering을 위해서 기존의 CNN모델을 텐서 space로 확장 시킨 모델
- PIL: 새로운 데이터에 대한 FCN의 파라메터 업데이트
  SIL: 새로운 데이터에 대한 모델 구조 업데이트

$$FT = f\left(I * K + b\right) \quad (1)$$

$$FT = f\left(\beta \cdot \text{pool}\left(T\right) + b\right) \quad (2)$$

$$H = f\left(W \odot I + b\right) \quad (3)$$

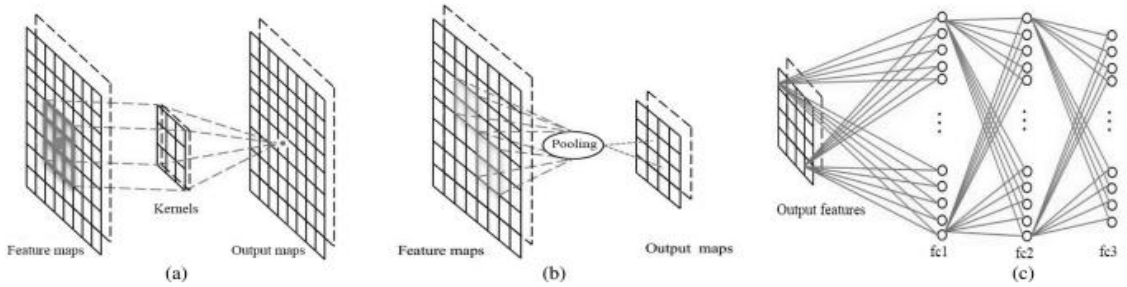


Fig. 1. Tensor computation of the deep convolutional computation model. (a) Tensor convolution. (b) Tensor pooling. (c) Fully connected.

Algorithm 1: High-order Back-Propagation Algorithm.

Input: Tensor example $\{(X^{(i)}, Y^{(i)})\}$, $step_{\max}$, Learning rate η, $threshold$

Output: Updated parameters $\theta = W, b, K, \beta$

Forward Pass:

Extract features

Tensor convolution $Map_c = f\left(X * K + b\right)$;

Tensor pooling $Map_s = f\left(\beta \cdot pool\left(Map_c\right) + b\right)$;

Map features and compute outputs

$H = f\left(W \odot Map_s + b\right)$;

Backprogation Process:

Differentiate loss $\Delta\delta_\theta$ with respect to $\theta = W, b, K, \beta$;

Update tensor fully connected layer

$W_f = W_f - \eta\nabla\delta_{\theta W_f}$;

Update tensor pooling layer $W_s = W_s - \eta\nabla\delta_{\theta W_s}$;

Update tensor pooling layer $W_c = W_c - \eta\nabla\delta_{\theta W_c}$;

INCREMENTAL DCCM

A. PIL 알고리즘

1) Initial increment computation
: 새로운 loss function

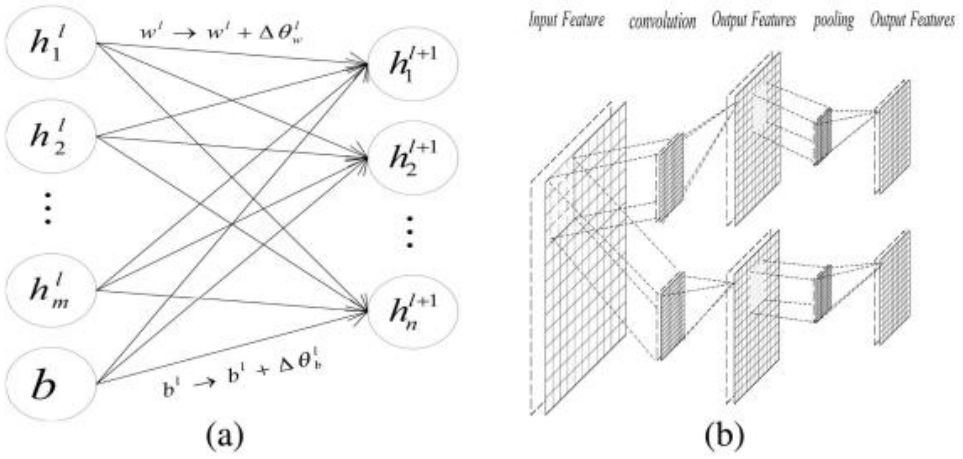


Fig. 2. Incremental DCCM. (a) Parameter computation method. (b) Basic constrained network.

Historical knowledge -> final value should the adaption and preservation

$$\theta = \{W^l, b^l | l = 1, 2, \ldots, L\}$$

$$\Delta\theta = \{\Delta W^l, \Delta b^l | l = 1, 2, \ldots, L\}$$

Initial parameter increment (speed up the training process를 목표로 함)

$$x_l = X_{i_1 i_2 \cdots i_N}, l = i_1 + \sum_{j=2}^{N} \prod_{k=1}^{j-1} I_k \qquad (5)$$

$$J_{\text{Increment}} = J_{\text{Adaption}} + J_{\text{Preservation}} \qquad (6)$$

Novel loss function = postprediction loss + deformation loss

$$J_{\text{Adaption}} = \frac{1}{2}\Delta y^T \Theta \Delta y, \Delta y = H_{\theta+\Delta\theta}(x) - y \qquad (7)$$

Target label과 updated parameter를 통한 network output의 차이

$$J_{\text{Preservation}} = \frac{1}{2}\left\| H_{\theta+\Delta\theta}(x) - H_\theta(x) \right\|_2^2. \qquad (8)$$

Updated output과 초기의 output사이의 L_2 -norm

$$\frac{\partial J}{\partial \Delta\theta} = \frac{\partial}{\partial \Delta\theta}\left(\frac{1}{2}\Delta y^T \Theta \Delta y + \frac{1}{2}\left\| H_{\theta+\Delta\theta}(x) - H_\theta(x) \right\|_2^2\right) = 0.$$

INCREMENTAL DCCM

A. PIL 알고리즘

2) Incremental training
: Dropout training(FCN에서의 idle 노드들이 새로운 정보를 습득하도록 학습) &
Final fine-tuning training

FCN의 tensor 뉴런들을 3개의 하위 집합으로 분류하였다. 각 하위 집합에 속한 뉴런들은 ***pi*** {i=1,2,3}(베르누이 분포를 따름)의 drop out 확률을 가진다. Idle한 뉴런들은 학습과정에서 높은 확률을 가지게 되고 과거 정보를 포함하는 뉴런은 낮은 확률을 가지게 된다.

$$\|\mathrm{w}\| = w \otimes w = \sum_{i_1=1}^{I_1} \cdots \sum_{i_{n-1}}^{I_{N-1}} w^2_{i_1 i_2 \ldots i_{N-1}}. \qquad (12)$$

다른 하위 집합에 속한 뉴런들은 feature 학습시 다른 역할을 수행한다. WA는 새로운 데이터 학습, WB는 대부분 지식 보존, Wc는 특정 example 정보 처리

Fine-tuning은 dropout없이 조정된 파라메터로 예측값을 계산한다

새로운 데이터를 학습하기
위해서는 WA에 속한 뉴런들이
높은 확률로 학습되어야함

$$W_A : 0 < \|\mathrm{w}\| < R_1$$

$$W_B : r_1 \le \|\mathrm{w}\| < R_2$$

$$W_C : \|\mathrm{w}\| \ge R_2. \qquad (13)$$

INCREMENTAL DCCM

B. SIL 알고리즘
: Rules for the model topology increment (Tensor convolutional, pooling, and FCN layers가 과거 정보를 transfer하도록 설계 & dropout extension into tensor space)

1) Increment of the FCN
: 구조적 통합을 위해서 0의 weight값을 가지는 virtual nodes가 도입되었다.
공식에 따른 Topology increment 이후에 정규분포를 따르는 난수에 의해서 virtual 노드를 제외한 weight tensor들은 초기화 된다.

$$N = L \times I_2 \times I_3 \cdots I_N - M$$

Virtual nodes

$$L = \lceil M \div (I_2 \times I_3 \cdots I_N) \rceil$$

$$w \in R^{\alpha \times I_1 \times I_2 \cdots I_N}, (\alpha = J_1 \times J_2 \cdots J_N)$$

$$b \in R^{J_1 \times J_2 \cdots J_N}.$$

$$w \in R^{\alpha \times (I_1 + L_1) \times I_2 \cdots I_N}, (\alpha = (J_1 + L_2) \times J_2 \cdots J_N)$$

$$b \in R^{(J_1 + L_2) \times J_2 \cdots J_N}.$$

M1, M2개의 new 노드가 input과 output layer에 추가되었을때

INCREMENTAL DCCM

B. SIL 알고리즘

2) Increment of the Constrained Network
: The incremental topology of the constrained network is mainly expressed by the increment of the tensor convolutional and pooling kernels. 하지만 기존의 커널은 기존 데이터의 유실없이 새로운 데이터에 대한 unknown feature 반영 불가. 네트워크 topology의 업데이트 후, 새로운 커널들은 정규분포를 따르는 0에 근사한 수로 초기화 된다.

$X \in R^{H_1 \times H_2 \cdots H_N}$ Input feature

$K_c \in R^{s \times I_1 \times I_2 \cdots I_N}$ Tensor convolutional subkernel

$K_p \in R^{s \times J_1 \times J_2 \cdots J_N}$ Tensor pooling subkernel

$O_c \in R^{s \times (H_1 - I_1 + 1) \times (H_1 - I_1 + 1) \cdots (H_N - I_N + 1)}$

$O_p \in R^{s \times \frac{H_1 - I_1 + 1}{J_1} \times \frac{H_2 - I_2 + 1}{J_2} \cdots \frac{H_N - I_N + 1}{J_N}}$

M new tensor convoluional subkernels join into the convolutional layer with M elements added in bias.

$K_c \in R^{(s+M) \times I_1 \times I_2 \cdots I_N}$

$K_p \in R^{(s+M) \times J_1 \times J_2 \cdots J_N}$

$O_c \in R^{(s+M) \times (H_1 - I_1 + 1) \times (H_1 - I_1 + 1) \cdots (H_N - I_N + 1)}$

$O_p \in R^{(s+M) \times \frac{H_1 - I_1 + 1}{J_1} \times \frac{H_2 - I_2 + 1}{J_2} \cdots \frac{H_N - I_N + 1}{J_N}}$.

3) Dropout High-Back Propagation algorithmn

EXPERIMENT

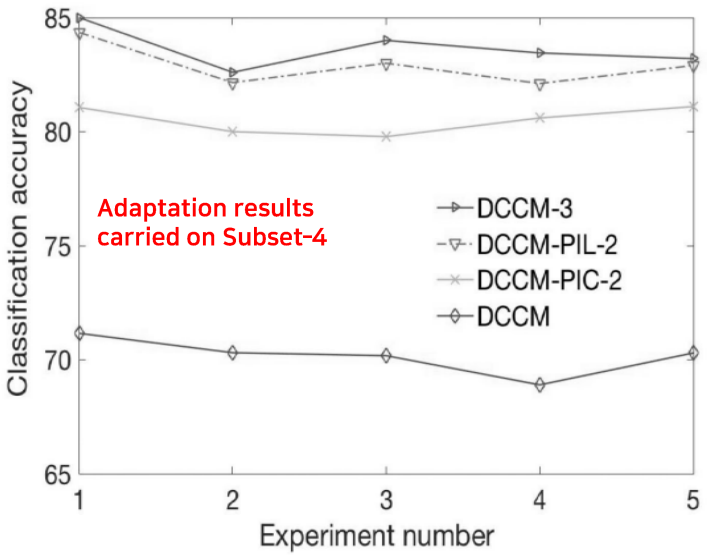
- 테스트 데이터는 총 5개의 subset으로 구성

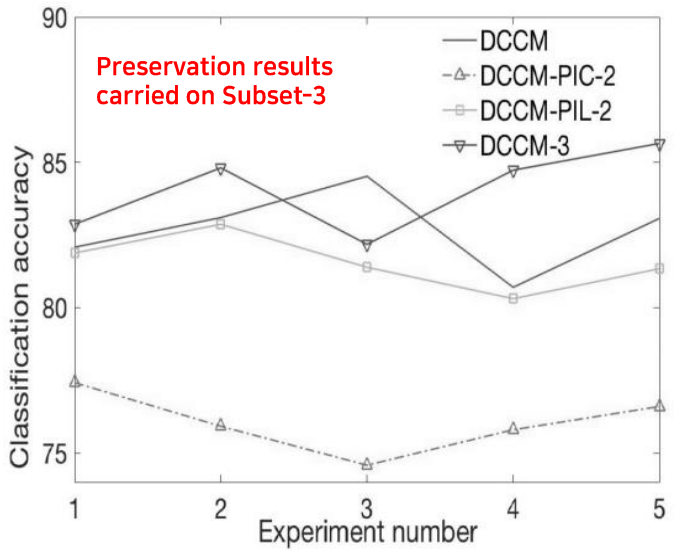
TABLE I
SHORT NAME OF THE MODELS USED IN CIFAR DATASET

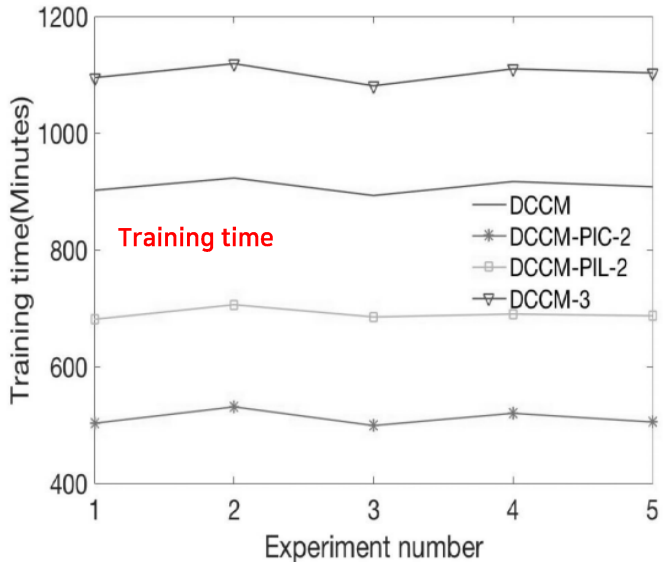
Model name	Description
DCCM	DCCM trained by HBP on the subset-1
DCCM-PIC-2	DCCM trained by PIC on the subset-2
DCCM-PIL-2	DCCM trained by PIL on the subset-2 and subset-1
DCCM-3	DCCM trained by HBP on the whole training data

PIC: PIDCCM과 비교 대상
PIL: proposed method

1) Subset-1 is the training dataset including four subclasses of each superclass.
2) Subset-2 is the incremental training dataset including the remaining subclass of each superclass.
3) Subset-3 is the preservation test dataset including the same subclasses with subset-1.
4) Subset-4 is the adaption test dataset including the same subclass with subset-2.







EXPERIMENT

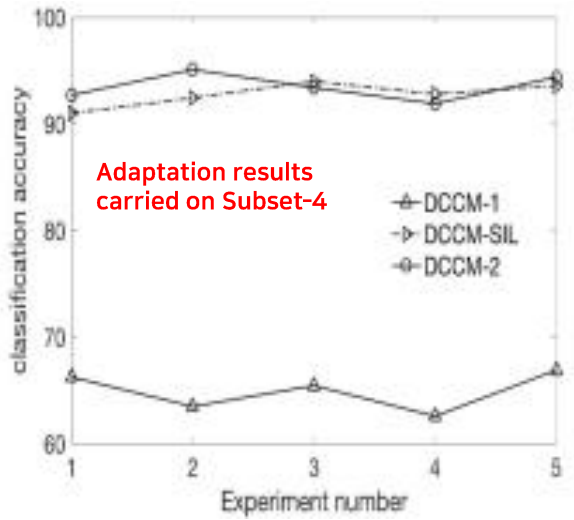
- 테스트 데이터는 총 5개의 subset으로 구성

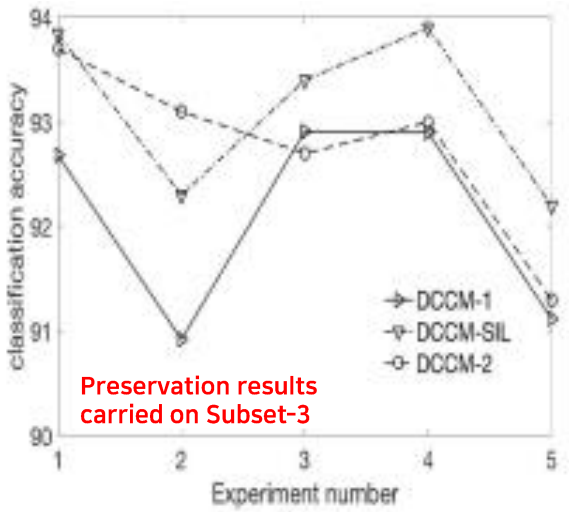
TABLE II
SHORT NAME OF THE MODELS USED IN CUAVE DATASET

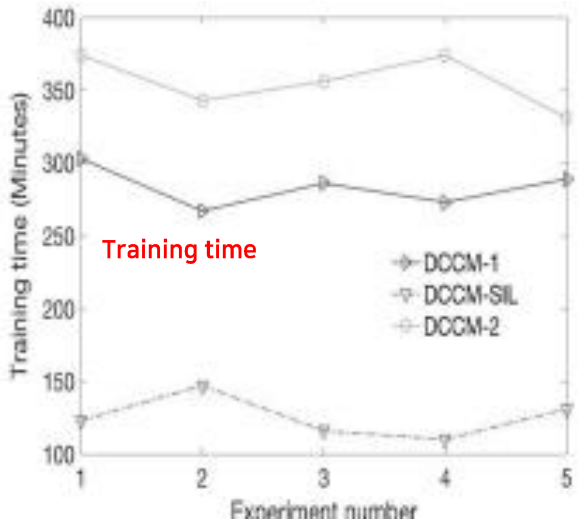
Model name	Description
DCCM-1	DCCM trained by HBP on the subset-1
DCCM-SIL	DCCM trained by SIL on the subset-1 subset-2
DCCM-2	DCCM trained by HBP on the subset-1 and subset-2

HBP: PIDCCM과 비교 대상
SIL: proposed method

1) Subset-1 is the initial training dataset including objects labeled with the digit from 0 to 7.
2) Subset-2 is the incremental dataset including the remaining objects.
3) Subset-3 is the preservation test dataset including objects of the same label subset-1.
4) Subset-4 is the adaption test dataset including objects of the same label subset-2.







IMPRESSION

- ✓ 정확도 손실 없이 모델 학습의 가속화를 위해서 *tensor decomposition* 활용 가능
  -> singular value decomposition on a fully connected layer
  -> convolution forward pass with CP Decomposition