FOSTER: Feature Boosting and Compression for Class-Incremental Learning

- Young Jo Choi
- Department of Digital Analytics

FOSTER: Feature Boosting and Compression for Class-Incremental Learning

Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye✉, and De-Chuan Zhan

State Key Laboratory for Novel Software Technology, Nanjing University
`wangfuyun@smail.nju.edu.cn,{zhoudw, yehj, zhandc}@lamda.nju.edu.cn`

Comments: Accepted to ECCV 2022. Code is available at: this https URL
Subjects: **Computer Vision and Pattern Recognition (cs.CV)**; Machine Learning (cs.LG)
Cite as: arXiv:2204.04662 **[cs.CV]**
(or arXiv:2204.04662v2 **[cs.CV]** for this version)
https://doi.org/10.48550/arXiv.2204.04662

Introduction

Incremental Learning

- The ability to continuously learn new data sets of new classes that were not seen in previous tasks.

- Many works have been proposed to alleviate catastrophic forgetting phenomenon, whereas most of them either fall into the stability-plasticity dilemma or take too much computation or storage overhead.

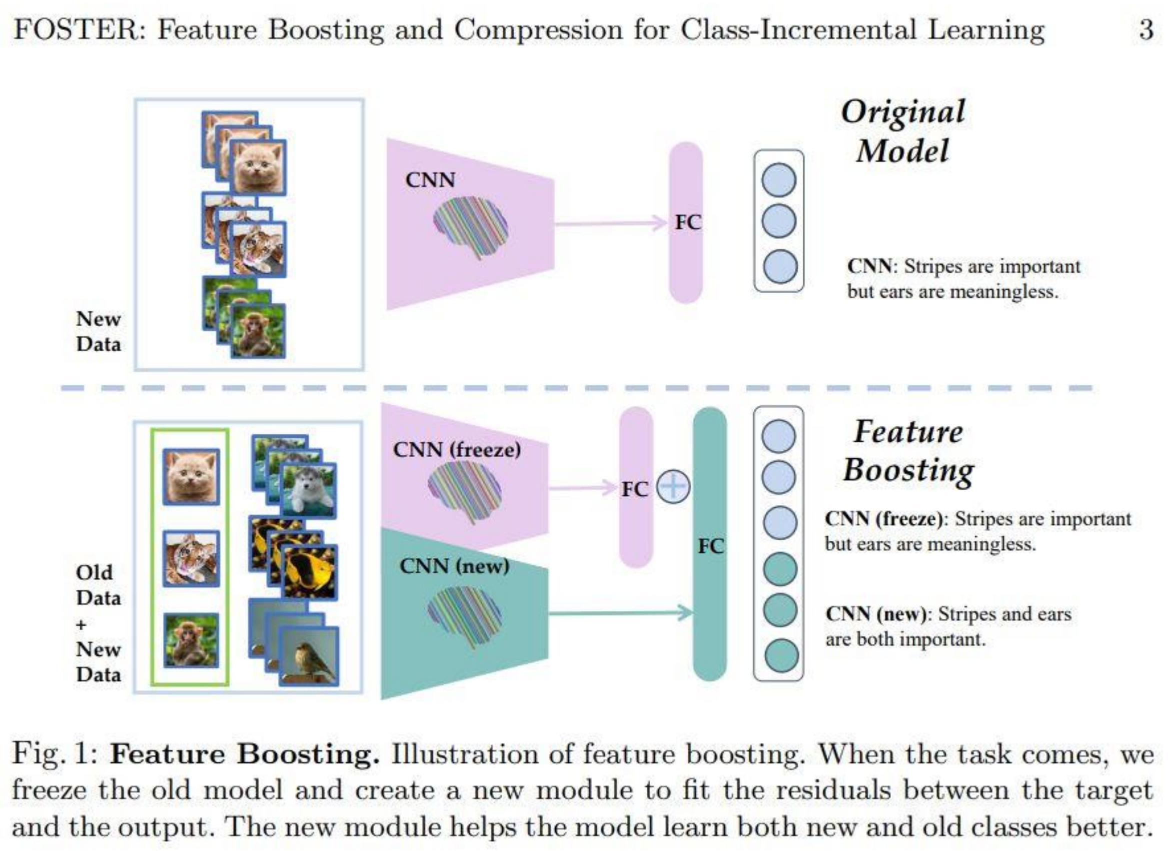
Introduction

Inevitable Defects

- First, constantly expanding new modules for coming tasks will lead to a drastic increase in the number of parameters, resulting in severe storage and computation overhead, which makes these methods not suitable for long-term incremental learning.

- Second, since old modules have never seen new concepts, directly retaining them may harm performance in new categories. The older modules kept, the more remarkable the negative impact.

Introduction



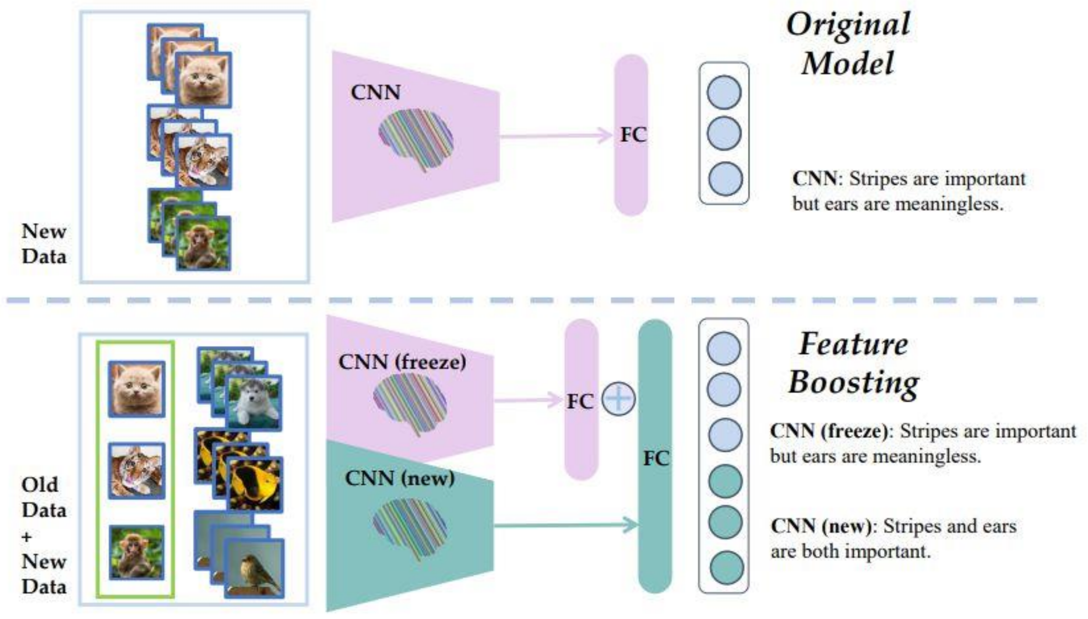

New
Data
CNN
FC
Original
Model
CNN: Stripes are important but ears are meaningless.
Old
Data
+
New
Data
CNN (freeze)
CNN (new)
FC
FC
Feature
Boosting
CNN (freeze): Stripes are important but ears are meaningless.
CNN (new): Stripes and ears are both important.

Fig. 1: **Feature Boosting.** Illustration of feature boosting. When the task comes, we freeze the old model and create a new module to fit the residuals between the target and the output. The new module helps the model learn both new and old classes better.

- Step1) **boosting**
  - Alleviate the performance decline due to the arrival of new classes.
- Step2) **compression**
  - Eliminate redundant parameters and meaningless dimension caused by feature boosting

Related Work

Knowledge Distillation

- Aims to transfer knowledge from the teacher to the student
- by encouraging the outputs of the student model to approximate the outputs of the teacher model.

Rehearsal

- Enables the model to have partial access to old data allocate a memory to store exemplars of previous tasks.

Boosting

- Represents a family of machine learning algorithms that convert weak learners to strong ones.

Related Work

Dynamic Architectures

- create new modules to handle the growing training distribution dynamically.

- However, they have two unavoidable shortcomings:

  (i) Continually adding new modules causes unaffordable overhead.

  (ii) Directly retaining old modules leads to noise in the representations of new categories, harming the performance in new classes.

Gradient boosting

- **objective**

$$F^* = \arg\min_{F} \mathbb{E}_{(x,y)\in\mathcal{D}_{train}}\left[\ell\left(y, F(x)\right)\right]$$

, $\ell(\cdot,\cdot)$: loss function

- **Expression**
  iteratively adding a week function $h_i(\cdot)$ with a_i (coefficient of $h_i(\cdot)$) to gradually fit residuals.

$$F(x) = F_m(x) = \sum_{i=1}^{m} \alpha_i h_i(x)$$

- **Find next optimization**

$$F_{m+1}(x) = F_m(x) + \arg\min_{h_{m+1}\in\mathcal{H}_{m+1}} \mathbb{E}_{(x,y)\in\mathcal{D}_{train}}\left[\ell\left(y, F_m(x) + h_{m+1}(x)\right)\right]$$

$\Rightarrow$ Directly optimizing the function to find the best h_{m+1} is typically infeasible.

Gradient boosting

- **The steepest descent step**

$$F_{m+1}(x) = F_m(x) - \alpha_m \nabla_{F_m} \mathbb{E}_{(x,y)\in\mathcal{D}_{train}} \left[\ell\left(y, F_m(x)\right)\right]$$

where $-\nabla_{F_m} \mathbb{E}_{(x,y)\in\mathcal{D}_{train}} \left[\ell\left(y, F_m(x)\right)\right]$ is the objective for $h_{m+1}(x)$

- **Transform**

if $\ell(\cdot,\cdot)$ is mean-squared error, objective of $h_{m+1}(x)$ transforms to

$$-\nabla_{F_m} \mathbb{E}_{(x,y)\in\mathcal{D}_{train}} \left[\left(y - F_m(x)\right)^2\right] = 2 \times \mathbb{E}_{(x,y)\in\mathcal{D}_{train}} \left[y - F_m(x)\right]$$

For each $(x, y) \in D_{train}$, F_{m+1} is the optimal function, minimizing the empirical error.

Class-Incremental Learning Setup

- new task set at t^{th} stage

  $$D_t = \{(x_i^t, y_i^t)\}_{i=1}^{n}$$

- the number of training samples : n
- input image and its corresponding label

  $$x_i^t \in X_t \ , \ y_i^t \in \mathcal{Y}_t$$

- label Space

  $$\hat{\mathcal{Y}}_t = \cup_{i=0}^{t} \mathcal{Y}_i \, ,$$

  $$where\ \mathcal{Y}_t \cap \mathcal{Y}_{t'} = \emptyset \ \ for\ \ t \neq t'$$

- rehearsal memory

  $$\mathcal{V}_t = a\ limited\ subset\ of\ \cup_{i=0}^{t-1} D_i$$

- training set

  $$\hat{D}_t = D_t \cup \mathcal{V}_t$$

From Gradient Boosting to Class-Incremental Learning

F_{t-1} is already trained at last stage,

$$\mathrm{F}_{t-1}(\boldsymbol{x}) = (\mathbf{W}_{t-1})^{\top}\Phi_{t-1}(\boldsymbol{x}), \text{ where } \Phi_{t-1}(\cdot) : \mathbb{R}^{D} \rightarrow \mathbb{R}^{d} \text{ and } \mathbf{W}_{t-1} \in \mathbb{R}^{d \times |\hat{\mathcal{Y}}_{t-1}|}$$

($\Phi_{t-1}(x)$: Feature Extractor for $x \in \widehat{D}_{t-1}$)

When a new data stream comes, directly fine-tuning F_{t-1}on the new data will **impair its capacity for old classes**. On the other hand, simply freezing F_{t-1} causes it to **lose plasticity** for new classes, making the residuals between target y and $F_{t-1}(x)$ large.

From Gradient Boosting to Class-Incremental Learning

Inspired by gradient boosting, we train a new model to fit the residuals.

At t^{th} stage, new model $\mathcal{F}_t$, consist of a feature extractor $\phi_t(\cdot) : \mathbb{R}^D \rightarrow \mathbb{R}^d$ and a linear classifier $W_t \in \mathbb{R}^{d\times|\hat{\mathcal{Y}}_t|}$.

W_t can be further decomposed into $\left[W_t^{(o)}, W_t^{(n)}\right], where\ W_t^{(o)} \in \mathbb{R}^{d\times|\hat{\mathcal{Y}}_{t-1}|}$ and $W_t^{(n)} \in \mathbb{R}^{d\times|\mathcal{Y}_t|}$.

From Gradient Boosting to Class-Incremental Learning

Training process expression

$$\mathrm{F}_t(\boldsymbol{x}) = \mathrm{F}_{t-1}(\boldsymbol{x}) + \underset{\mathcal{F}_t}{\arg\min}\, \mathbb{E}_{(\boldsymbol{x},y)\in\hat{\mathcal{D}}_t}\left[\ell\left(y, \mathrm{F}_{t-1}(\boldsymbol{x}) + \mathcal{F}_t(\boldsymbol{x})\right)\right]$$

$$\boldsymbol{y} = \mathrm{F}_{t-1}(\boldsymbol{x}) + \mathcal{F}_t(\boldsymbol{x}) = \mathcal{S}\left(\begin{bmatrix}\mathbf{W}_{t-1}^{\top}\\ \mathbf{O}\end{bmatrix}\Phi_{t-1}(\boldsymbol{x})\right) + \mathcal{S}\left(\begin{bmatrix}(\mathcal{W}_t^{(o)})^{\top}\\ (\mathcal{W}_t^{(n)})^{\top}\end{bmatrix}\phi_t(\boldsymbol{x})\right)$$

$S(\cdot)$: the softmax function
$\boldsymbol{O} \in \mathbb{R}^{d\times|\mathcal{Y}_t|}$: set to zero matrix or finetuned on $\widehat{D}_t$ with $\Phi_{t-1}(x)$ frozen.

θ_t : parameter of $\mathcal{F}_t$
$Dis(\cdot,\cdot)$: distance metric (e.g., Euclidean metric)

From Gradient Boosting to Class-Incremental Learning

$$\theta_t^* = \arg\min_{\theta_t} \mathrm{Dis}\left(\boldsymbol{y}, \mathcal{S}\left(\begin{bmatrix}\mathbf{W}_{t-1}^\top \\ \mathbf{O}\end{bmatrix} \Phi_{t-1}(\boldsymbol{x})\right) + \mathcal{S}\left(\begin{bmatrix}(\mathcal{W}_t^{(o)})^\top \\ (\mathcal{W}_t^{(n)})^\top\end{bmatrix} \phi_t(\boldsymbol{x})\right)\right)$$

Replace $S(\cdot) + S(\cdot)$ with $S(\cdot\ +\ \cdot)$, and substitute the $Dis(\cdot,\cdot)$ for the Kullback-Leibler divergence (KLD)

$$\Rightarrow \quad \theta_t^* = \arg\min_{\theta_t} \mathrm{KL}\left(y \,\middle\|\, \mathcal{S}\left(\begin{bmatrix}\mathbf{W}_{t-1}^\top & (\mathcal{W}_t^{(o)})^\top \\ \mathbf{O} & (\mathcal{W}_t^{(n)})^\top\end{bmatrix} \begin{bmatrix}\Phi_{t-1}(\boldsymbol{x}) \\ \phi_t(\boldsymbol{x})\end{bmatrix}\right)\right)$$

- Kullback-Leibler divergence (출처 : 위키백과)

$$D_{\mathrm{KL}}(P\|Q) = H(P,Q) - H(P)$$
$$= (-\sum_x p(x)\log q(x)) - (-\sum_x p(x)\log p(x))$$

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x)\log\left(\frac{p(x)}{q(x)}\right) dx$$

두 분포 P, Q의 cross entropy – P의 entropy

From Gradient Boosting to Class-Incremental Learning

F_t can be further decomposed → W_t and $\Phi_t(\cdot)$

$$\mathbf{W}_t^\top = \begin{bmatrix} \mathbf{W}_{t-1}^\top & (\mathcal{W}_t^{(o)})^\top \\ \mathbf{O} & (\mathcal{W}_t^{(n)})^\top \end{bmatrix} \qquad \Phi_t(\boldsymbol{x}) = \begin{bmatrix} \Phi_{t-1}(\boldsymbol{x}) \\ \phi_t(\boldsymbol{x}) \end{bmatrix}$$

Logits of F_t :

$$\mathbf{W}_t^\top \Phi_t(\boldsymbol{x}) = \begin{bmatrix} \mathbf{W}_{t-1}^\top \Phi_{t-1}(\boldsymbol{x}) + (\mathcal{W}_t^{(o)})^\top \phi_t(\boldsymbol{x}) \\ (\mathcal{W}_t^{(n)})^\top \phi_t(\boldsymbol{x}) \end{bmatrix}$$

For old classes

For lower classes

- Lower part requires the new module $\mathcal{F}_t$ to learn how to correctly classify new classes, thus enhancing the model's plasticity to redeem the performance on new classes.
- Upper part encourages the new module to fit the residuals between y and F_{t-1} , thus encouraging $\mathcal{F}_t$ to exploit more pivotal patterns for classification.

Calibration for Old and New

- When training on new tasks, we only have an imbalanced training set $\widehat{D}_t = D_t \cup \mathcal{V}_t$.
- The boosting model tends to ignore the residuals of minor classes due to insufficient supervision.

- To alleviate the classification bias and encourage the model to equally learn old and new classes, we propose Logits Alignment and Feature Enhancement strategies.

Calibration for Old and New

Logit Alignment

- We Add a scale factor to the logits of the old and new classes

$$\gamma \mathbf{W}_t^\top \Phi_t(\boldsymbol{x}) = \begin{bmatrix} \gamma_1 \left(\mathbf{W}_{t-1}^\top \Phi_{t-1}(\boldsymbol{x}) + (\mathcal{W}_t^{(o)})^\top \phi_t(\boldsymbol{x}) \right) \\ \gamma_2 (\mathcal{W}_t^{(n)})^\top \phi_t(\boldsymbol{x}) \end{bmatrix} \quad ,0 < \gamma_1 < \gamma_2$$

- γ_1 : small → absolute value of logits for old classed is reduced,
  γ_2 : large → absolute value of logits for old classed is enlarged

→ model will produce larger logits for old classes,
model will produce smaller logits for new classes.

Calibration for Old and New

Logit Alignment

We get γ_1 , γ_2 through the normalized number E_n of each class.

$$E_n = \begin{cases} \frac{1-\beta^n}{1-\beta}, & \beta \in [0,1) \\ n, & \beta = 1 \end{cases}$$

$$\Rightarrow \quad (\gamma_1, \gamma_2) = \left(\frac{E_{n_{\text{old}}}}{E_{n_{\text{old}}} + E_{n_{\text{new}}}}, \frac{E_{n_{\text{new}}}}{E_{n_{\text{old}}} + E_{n_{\text{new}}}} \right)$$

$$\Rightarrow \quad \mathcal{L}_{LA} = \mathrm{KL}\left(y \,\|\, \mathcal{S}\left(\gamma \mathbf{W}_t^\top \Phi_t(\boldsymbol{x})\right)\right)$$

Calibration for Old and New

Feature Enhancement

- At the extreme, for instance, the residuals of F_{t-1} and insufficient label y is zero.
- In that case, the new module $\mathcal{F}_t$ can not learn anything about old categories, and thus it will damage the performance for old classes.

→ prompt the new module $\mathcal{F}_t$ to learn old categories further.

Calibration for Old and New

Feature Enhancement

- Initialize new linear classifier $W_t^{(a)} \in \mathbb{R}^{d \times |\hat{y}_t|}$ to transform the new feature $\phi_t(x)$ into logits of all seen categories

- This requires the new feature itself to correctly classify all of them.

$$\mathcal{L}_{FE} = \mathrm{KL}\left(y \,\middle\|\, \mathcal{S}\left((\mathbf{W}_t^{(a)})^\top \phi_t(\boldsymbol{x})\right)\right)$$

Calibration for Old and New

Distillation

- To train the new feature extractor in an imbalanced dataset
  → failing to learn a feature representation with good generalization ability for old categories.

- To alleviate this phenomenon and provide more supervision for old classes, we utilize knowledge distillation to encourage $F_t(x)$ to have similar output distribution as F_{t-1} on old categories

$$\mathcal{L}_{KD} = \mathrm{KL}\left(\mathcal{S}\left(\mathrm{F}_{t-1}(\boldsymbol{x})\right) \middle\| \mathcal{S}\left(\mathrm{F}_{t-1}(\boldsymbol{x}) + (\mathcal{W}_t^{(o)})^\top \phi_t(\boldsymbol{x})\right)\right) \Rightarrow \mathcal{L}_{Boosting} = \mathcal{L}_{LA} + \mathcal{L}_{FE} + \mathcal{L}_{KD}$$

Feature Compression

(To avoid memory and storage problem)

FOSTER: Feature Boosting and Compression for Class-Incremental Learning 9

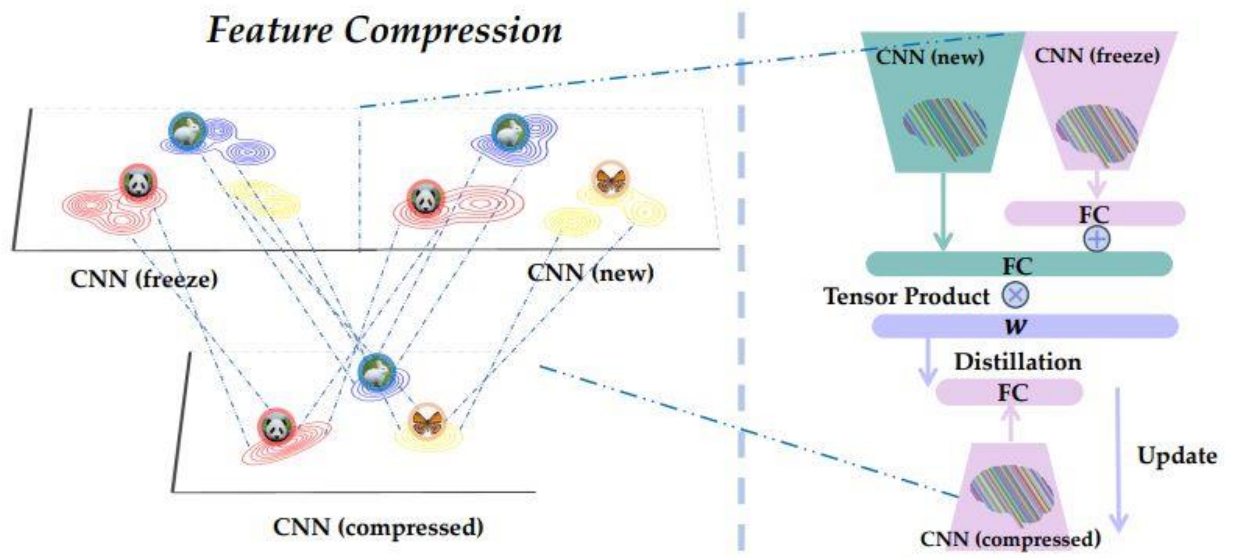


Fig. 2: **Feature Compression.** Left: the process of feature compression. We remove insignificant dimensions and parameters to make the distribution of the same categories more compact. Right: the implementation of feature compression. Outputs of the dual branch model are used to instruct the representation learning of the compressed model. Different weights are assigned to old and new classes to alleviate the classification bias.

Feature Compression

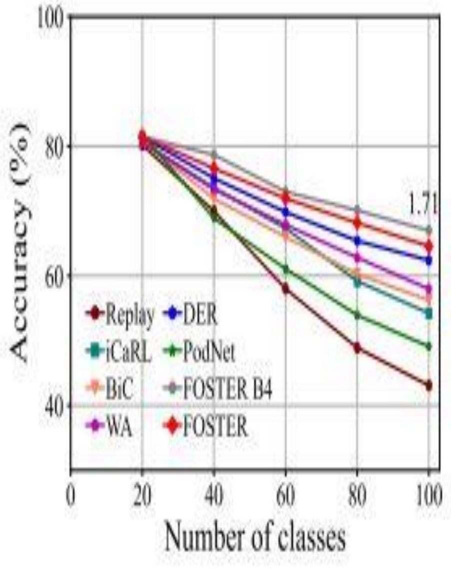
Balanced Distillation

- $F_t^{(s)}$: single backbone student to be distilled.
- To consider imbalance in train set,
  we adjust the weights of distilled information for different classes.

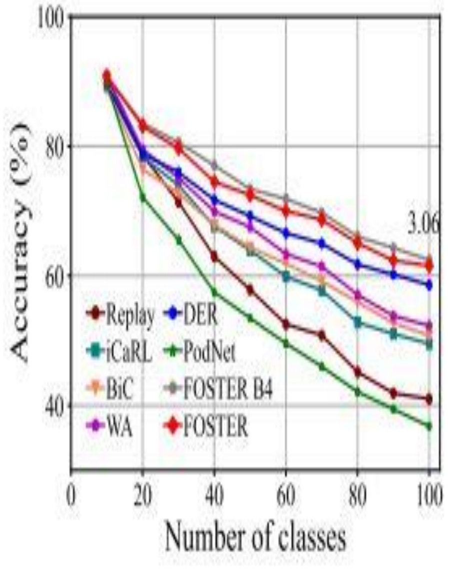
$$\mathcal{L}_{\mathrm{BKD}} = \mathrm{KL}\Big(\boldsymbol{w} \otimes \mathcal{S}\left(\mathrm{F}_t(\boldsymbol{x})\right) \,\Big\|\, \mathcal{S}(\mathrm{F}_t^{(s)}(\boldsymbol{x}))\Big)$$

- w : weighted vector obtained from $E_n = \begin{cases} \frac{1-\beta^n}{1-\beta}, & \beta \in [0,1) \\ n, & \beta = 1 \end{cases}$ to make classed with fewer instances have larger weights

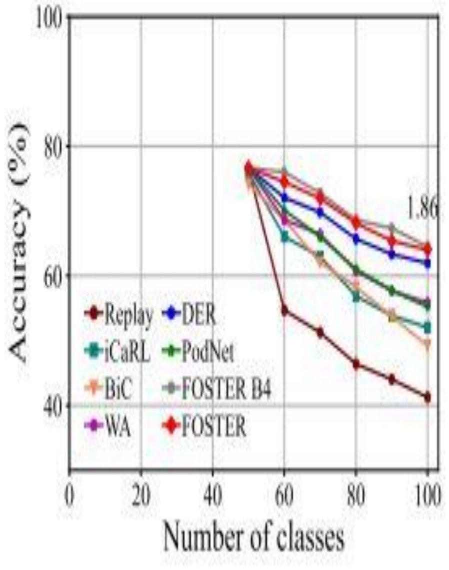
Experiments



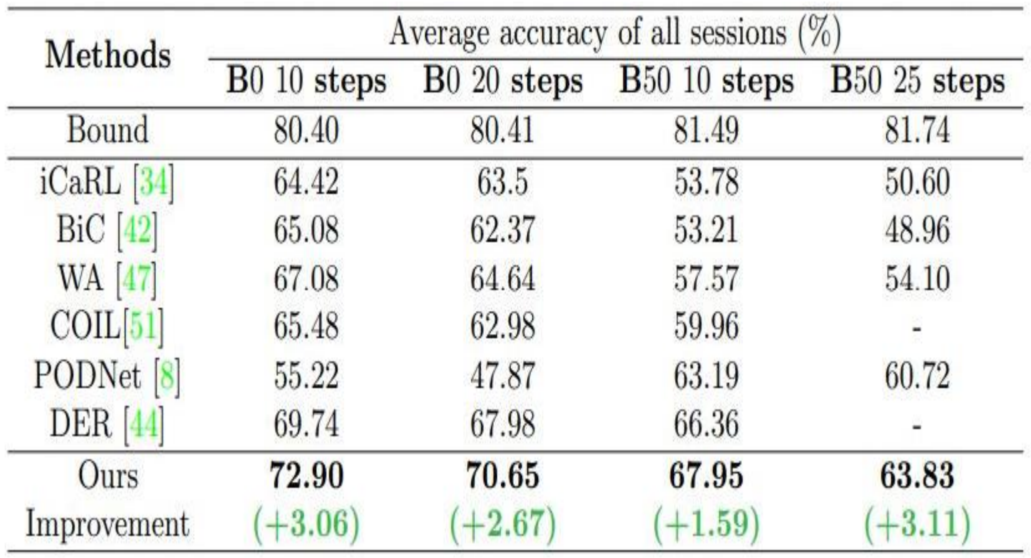
(a) CIFAR-100 B0 5 steps



(b) CIFAR-100 B0 10 steps



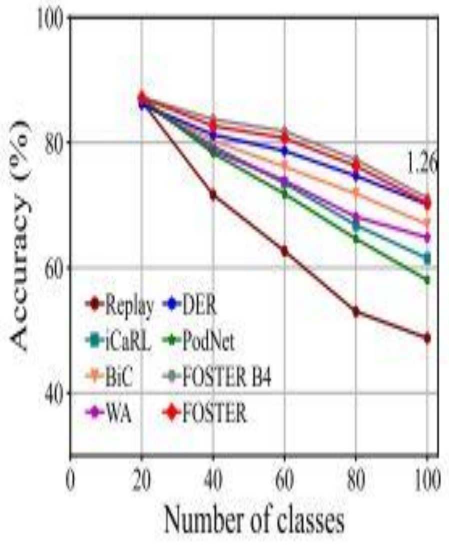
(c) CIFAR-100 B50 5 steps



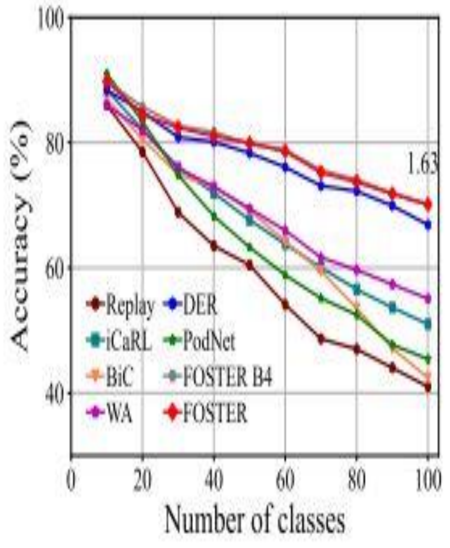
Methods	Average accuracy of all sessions (%)			
	B0 10 steps	B0 20 steps	B50 10 steps	B50 25 steps
Bound	80.40	80.41	81.49	81.74
iCaRL [34]	64.42	63.5	53.78	50.60
BiC [42]	65.08	62.37	53.21	48.96
WA [47]	67.08	64.64	57.57	54.10
COIL[51]	65.48	62.98	59.96	-
PODNet [8]	55.22	47.87	63.19	60.72
DER [44]	69.74	67.98	66.36	-
Ours	**72.90**	**70.65**	**67.95**	**63.83**
Improvement	(+3.06)	(+2.67)	(+1.59)	(+3.11)

- FOSTER B4 : dual branch model after feature boosting
- FOSTER : single backbone model after feature compression

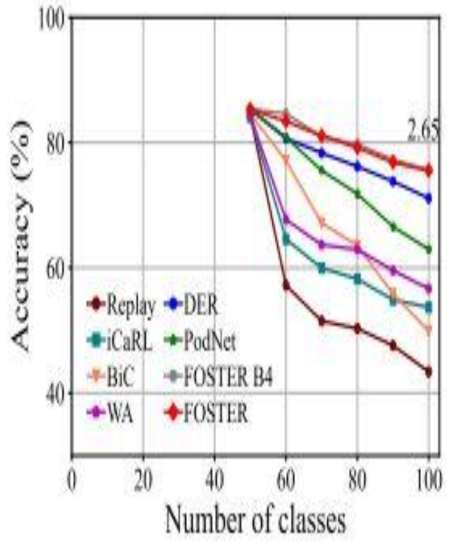
Experiments



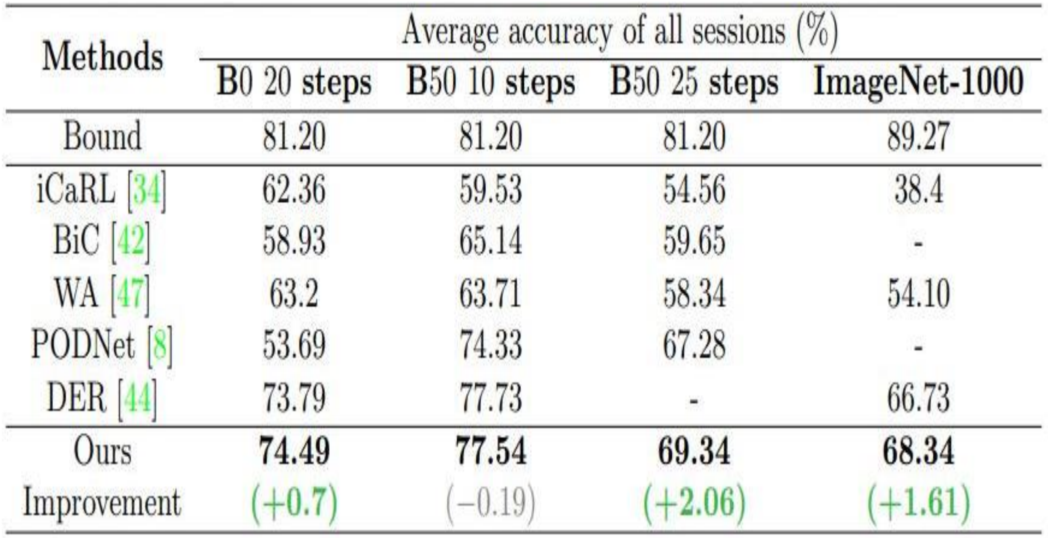
(a) ImageNet-100 B0 5 steps



(b) ImageNet-100 B0 10 steps



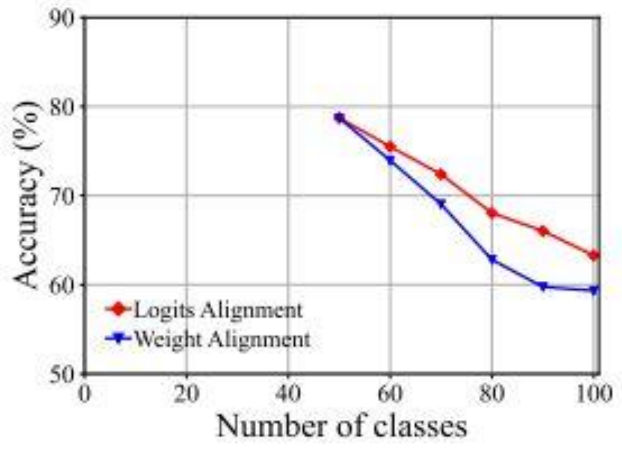
(c) ImageNet-100 B50 5 steps



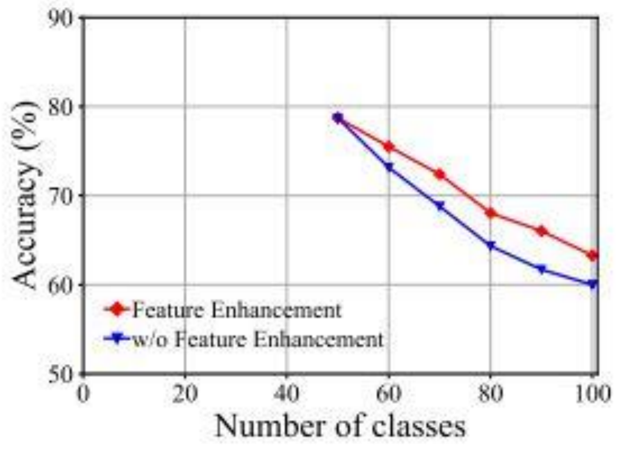
Methods	Average accuracy of all sessions (%)			
	B0 20 steps	B50 10 steps	B50 25 steps	ImageNet-1000
Bound	81.20	81.20	81.20	89.27
iCaRL [34]	62.36	59.53	54.56	38.4
BiC [42]	58.93	65.14	59.65	-
WA [47]	63.2	63.71	58.34	54.10
PODNet [8]	53.69	74.33	67.28	-
DER [44]	73.79	77.73	-	66.73
Ours	**74.49**	**77.54**	**69.34**	**68.34**
Improvement	(+0.7)	(−0.19)	(+2.06)	(+1.61)

- FOSTER B4 : dual branch model after feature boosting
- FOSTER : single backbone model after feature compression

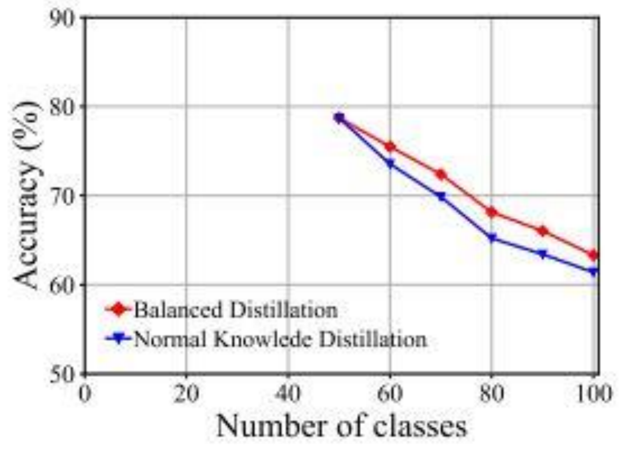
Ablation Study



(a) Logits alignment



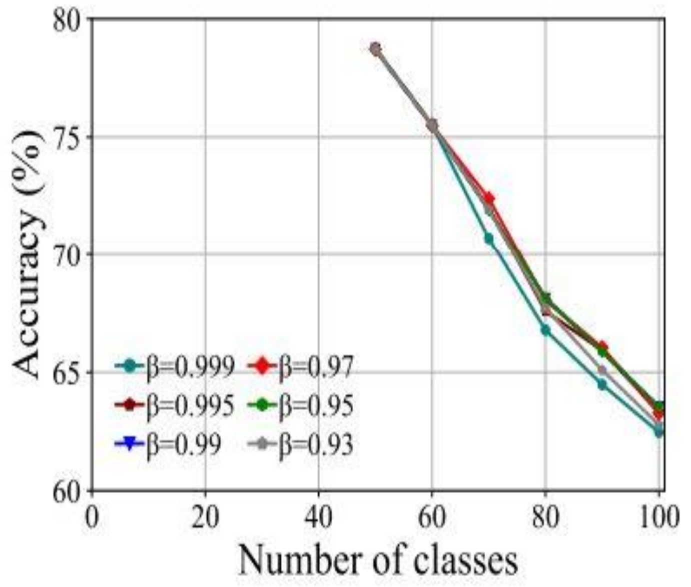
(b) Feature enhancement



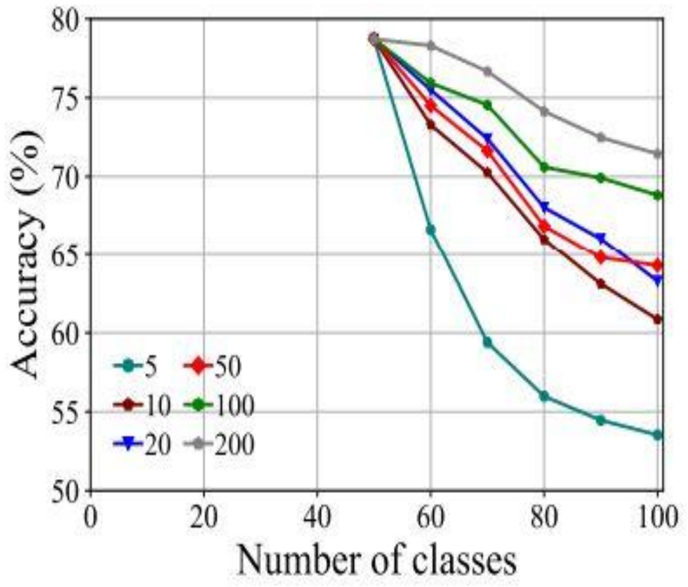
(c) Balanced distillation

Fig. 5: **Ablations** of the different key components of FOSTER. (a): Performance comparison between logits alignment and weight alignment [47]. (b): Performance comparison with or without Feature Enhancement. (c): Performance comparison between balanced distillation and normal knowledge distillation [19].

Ablation Study



(a) Sensitive study of hyper-parameters



(b) Influence of number of exemplars

Fig. 6: **Robustness Testing.** Left: Performance under different hyperparameter βs. Right: Performance with different numbers of exemplars. Both of them are evaluated on CIFAR-100 B50 with 5 steps.

Visualize Grad-CAM

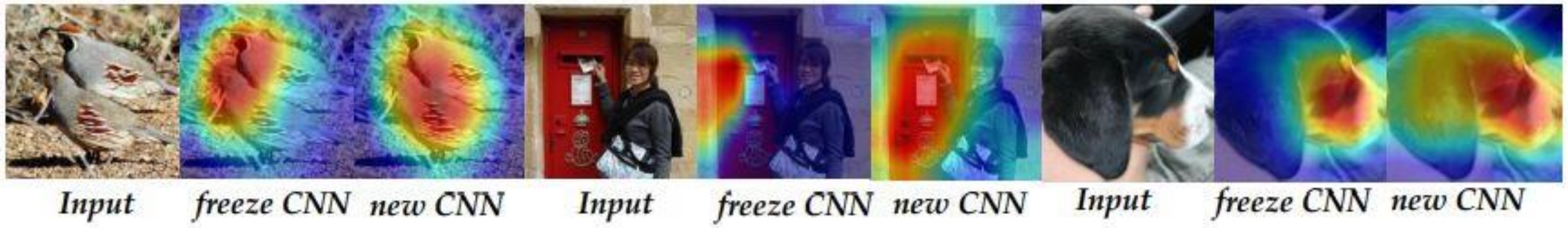


Fig. 7: **Grad-CAM before and after feature boosting.** The freeze CNN only focuses on some areas of an object and is not accurate enough, but the new CNN can discover those important but ignored patterns and correct the original output.

감사합니다

Introduction

Boosting

- In class-incremental learning, the distribution drift of new classes will also lead to the residuals of model.

- Therefore, we propose a boosting framework to solve the problem of class-incremental learning by applying an additive model, gradually fitting residuals, where different models mainly handle their special tasks (with nonoverlapping sets of classes).