Model Updates without Re-indexing: A Tutorial on Backward-Compatible Representations

Niccolò Biondi

Simone Ricci

Federico Pernici

Alberto Del Bimbo

Università degli Studi di Firenze, Italy

ICIAP2025, Rome

ICIAP 2025 ROME

MICC
Media Integration and Communication Center

Outline of the Tutorial

Compatibility Learning: The Context

- ✓ Visual Search Systems Updating
- ✓ Backfiling (re-indexing)
- ✓ Compatibility Learning

Compatibility Learning: The Solutions

- ✓ Using Regularization
- ✓ Using Mapping
- ✓ Using Architecture Modifications
- ✓ Using Softmax Outputs and Logits

Compatibility in Continual Learning

Compatibility with Foundation Models

- ✓ User-side Compatibility
- ✓ Third party-side Compatibility
- ✓ Negative Flips Reduction

Tutorial Presentation Schedule

September 16, 2025
2.30 pm – 6.00 pm

Part I 1.30h

COMPATIBILITY LEARNING: THE CONTEXT

- ✓ Visual Search Systems Updating
- ✓ Backfiling (re-indexing)
- ✓ Compatibility Learning

COMPATIBILITY LEARNING: THE SOLUTIONS

- ✓ Using Regularization
- ✓ Using Mapping
- ✓ Using Architectural Modifications

 **Coffee Break**

Part II 1.30h

COMPATIBILITY LEARNING: THE SOLUTIONS

- ✓ Using Softmax Outputs and Logits

Looking at the code

COMPATIBILITY IN CONTINUAL LEARNING

COMPATIBILITY WITH FOUNDATION MODELS

- ✓ User-side Compatibility

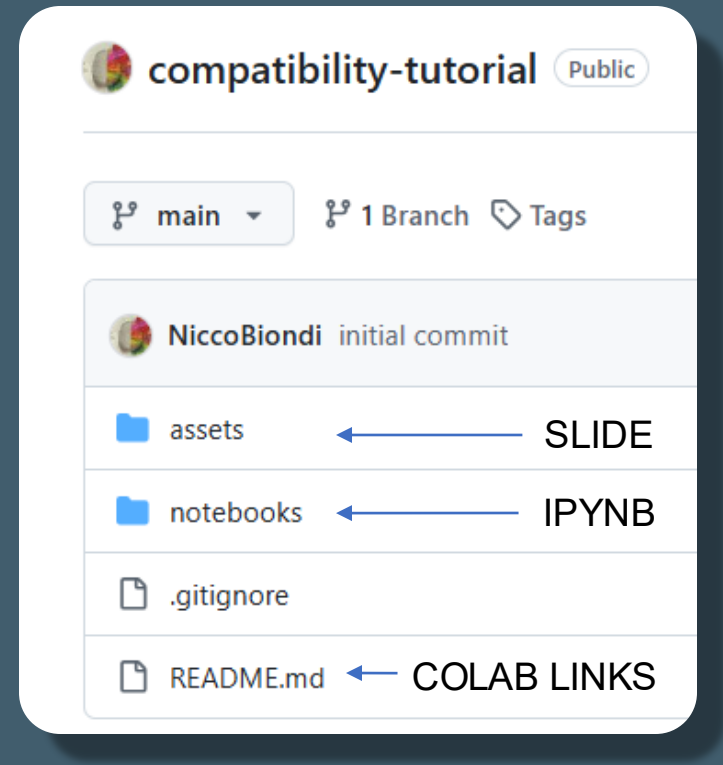
Looking at the code

- ✓ Third party-side Compatibility
- ✓ Negative Flips Reduction

GitHub Repo

You can get Slide and Practical Examples from our GitHub repository

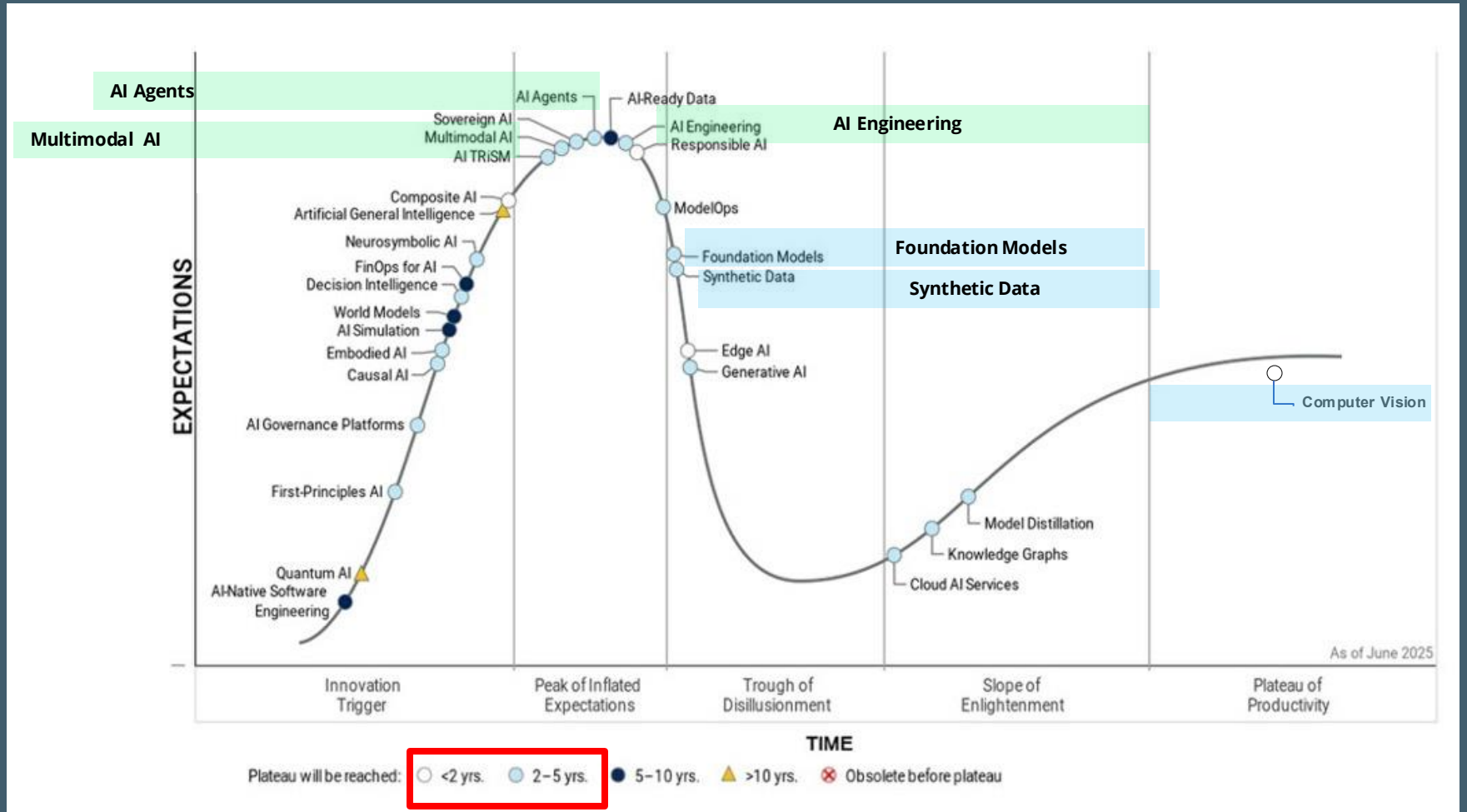
github.com/NiccoBiondi/compatibility-tutorial



Artificial Intelligence Y2025

« Artificial Intelligence is intelligence demonstrated by machines »

Wikipedia



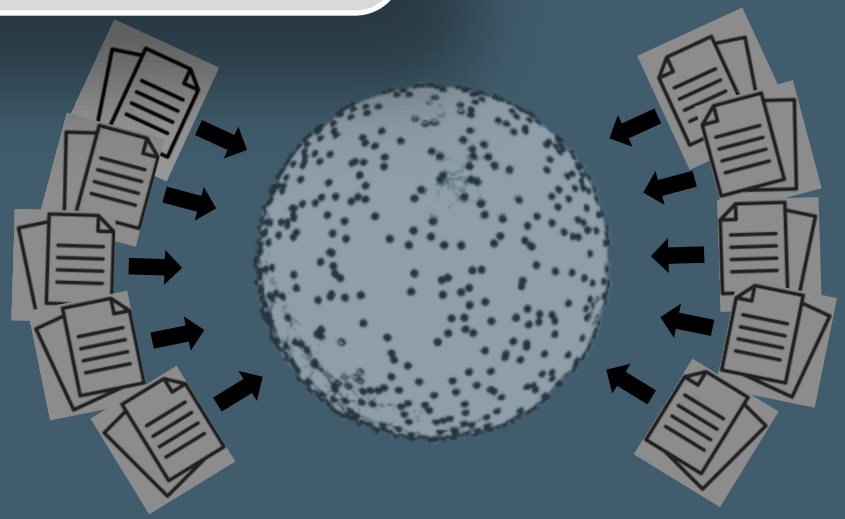
The Gartner's 2025 AI Hype Cycle: from peak of inflated expectations to plateau of productivity

Credits Gartner Inc. 2025 (modified)

Looking for Data

Training ML models require extensive, high-quality data

- ✓ Data collected from existing databases
- ✓ Data from web scraping:
  - Social Networks: more than 3.2 billion images and 720,000 hours of video shared daily
  - Common Crawl database includes 250 billion website pages and is expanded with 3–5 billion new pages each month
- ✓ Data generated during system use
- ✓ Data generated through simulation, data augmentation, or generative models



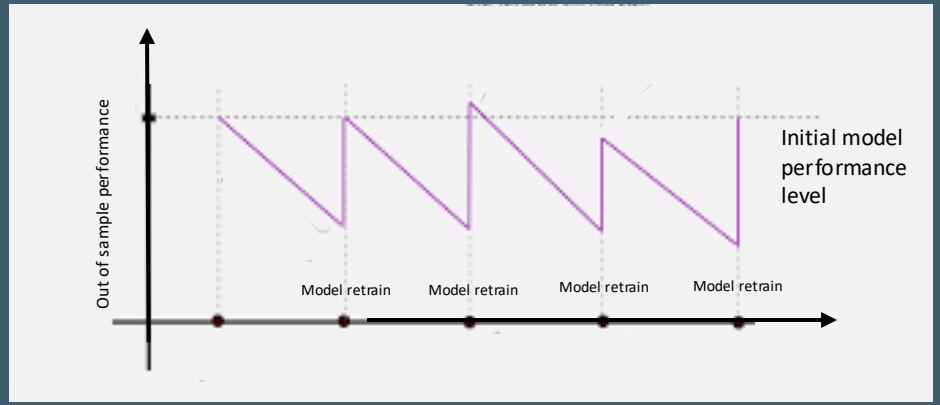
Model Updating

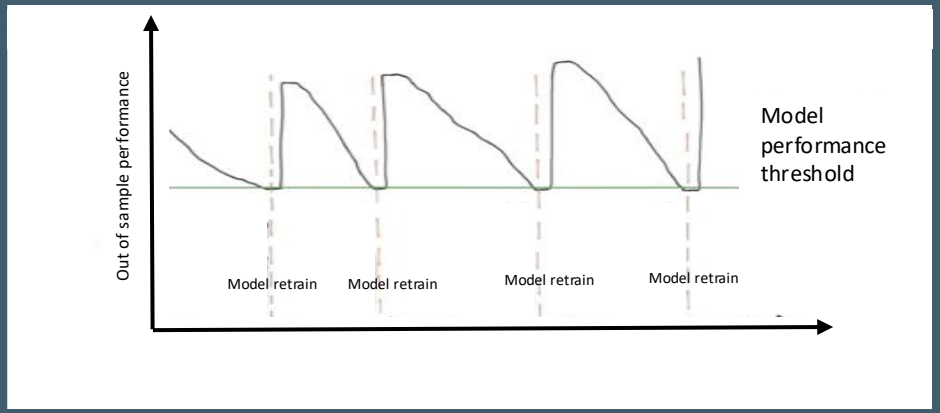
In real-world applications novelties constantly emerge: new data, new network architectures, different training techniques….

Models are released and updated continuously
to provide improved performance and a better user experience

Depending on the use case, the approaches for retraining a model include:

- ✓ Periodic retraining: the model is retrained at a specified time interval
- ✓ Trigger-based retraining: the model can be retrained automatically when its performance drops below a threshold

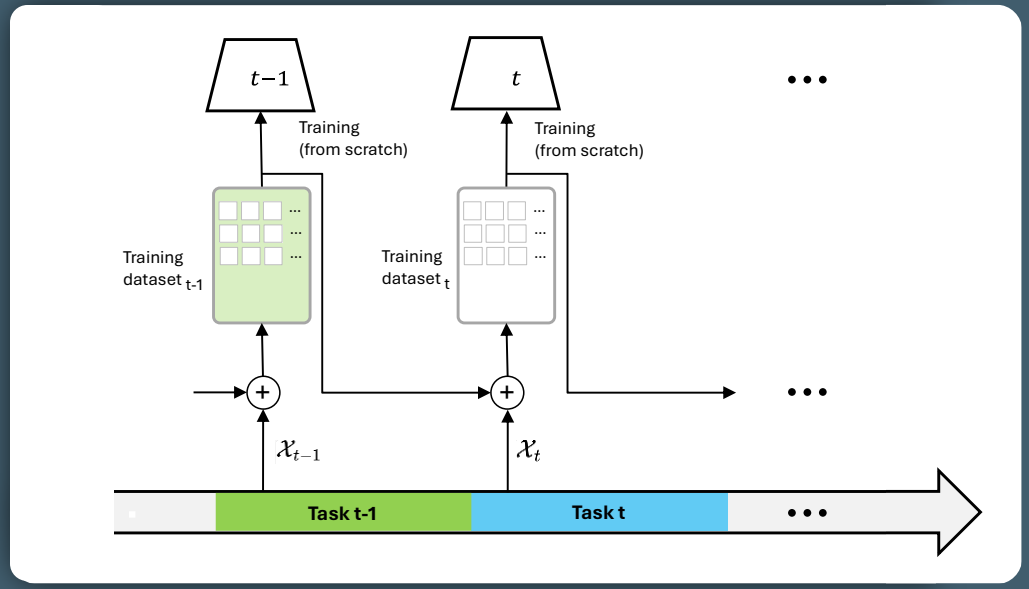


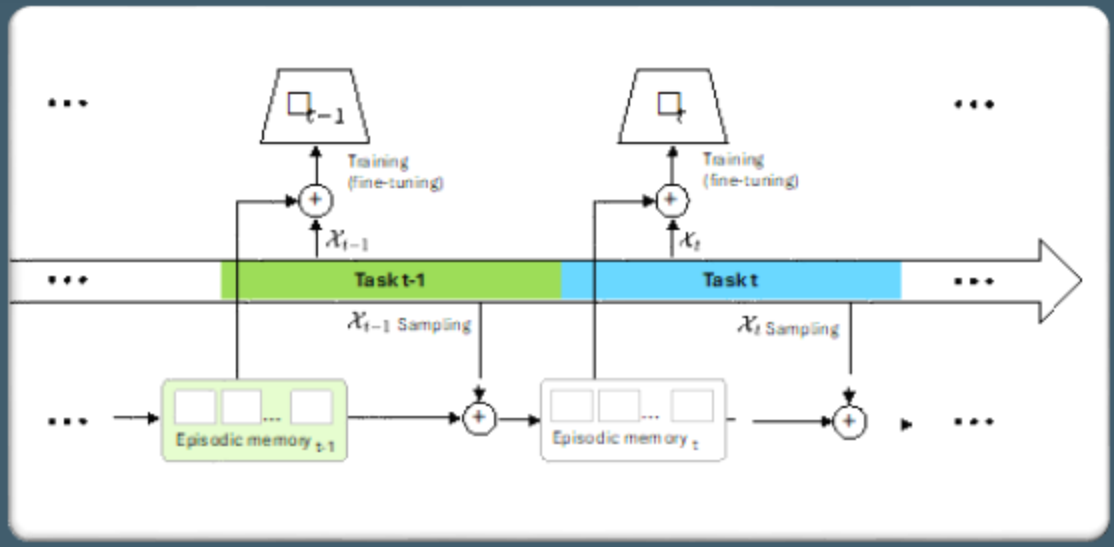


Retraining Machine Learning Models

Two primary retraining strategies:

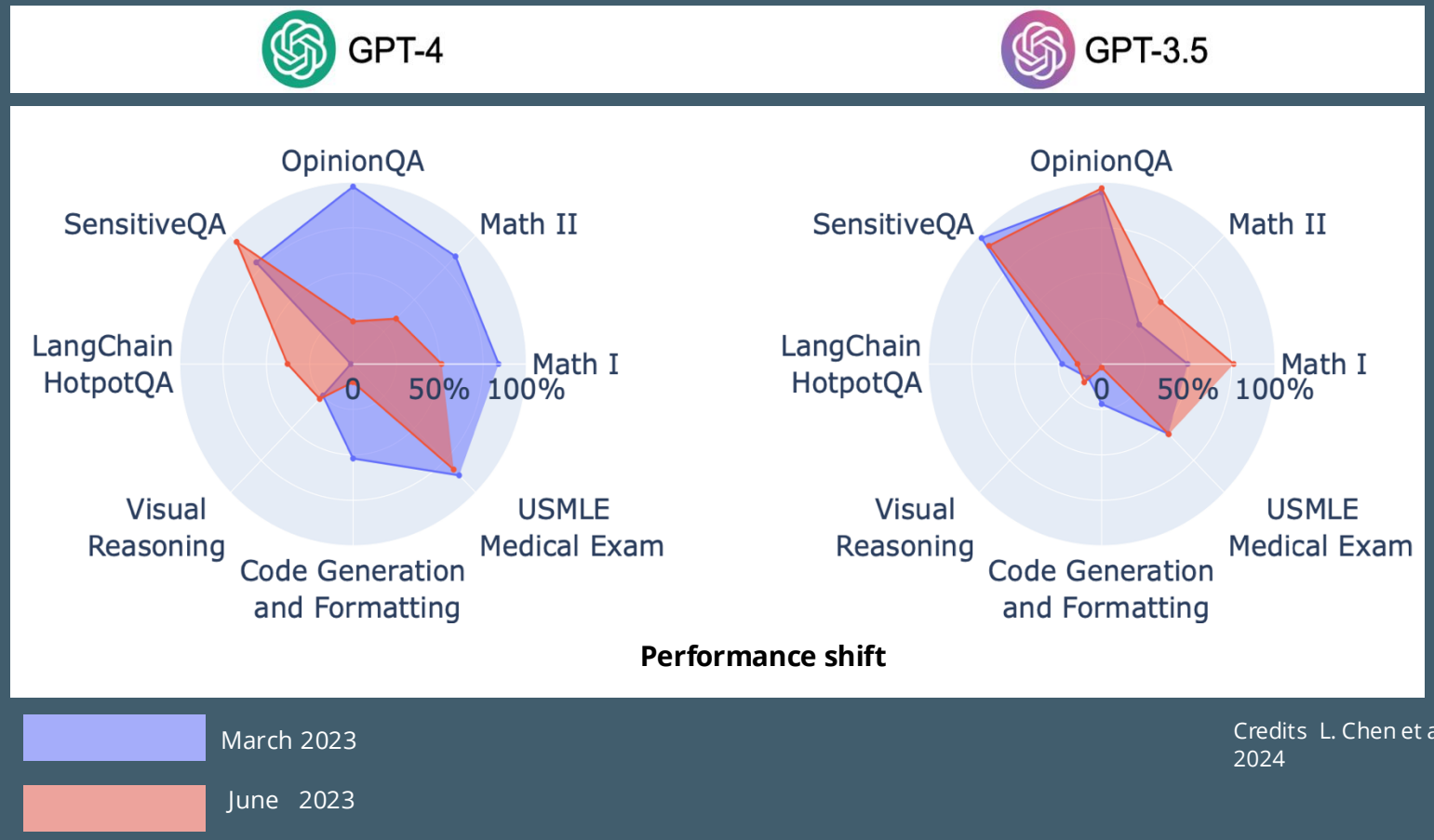
- ✓ **Full retraining:** involves rebuilding the model from scratch using both old and new data, ensuring complete learning but at a higher computational cost
- ✓ **Incremental learning:** allows models to update efficiently using only new data, making it ideal for large-scale or streaming applications





Model Updating: behavior change

In large models, updates to the model aimed at improving some aspects can reduce its capability in other dimensions

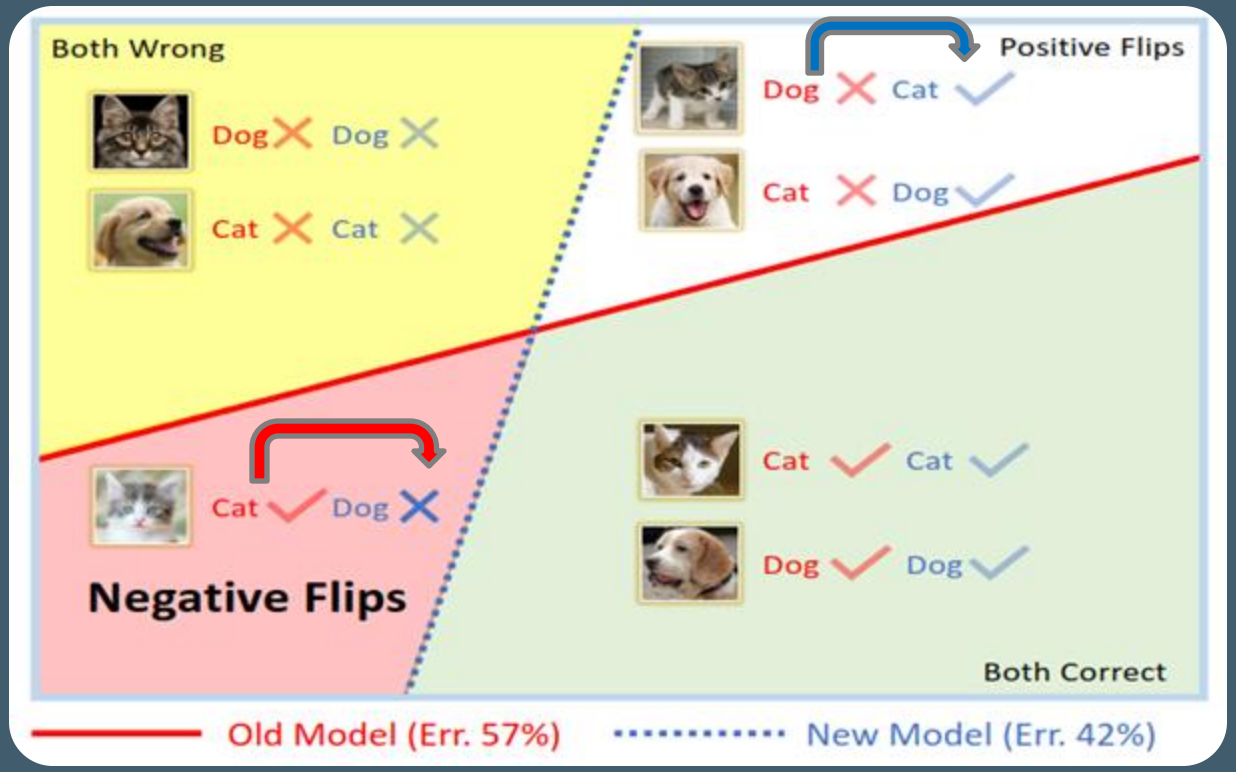


Credits L. Chen et al. 2024

Model Updating: behavior change

An update to a downstream task model could produce a worse prediction for many samples (**Negative flips**) even when it has a better overall performance wrt the old model

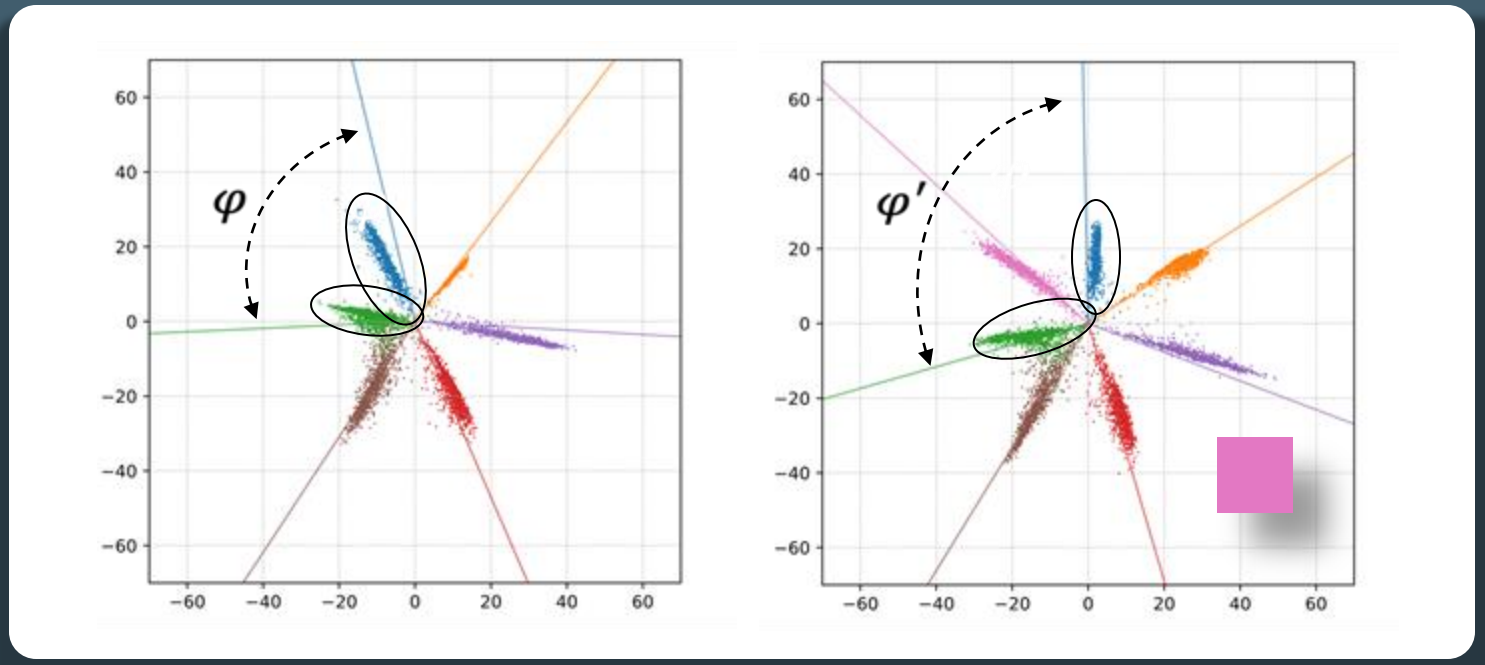
Picture authored by S.Yan



Model Updating: behavior change

Updating a model with **novel classes** changes the internal feature representation.
The geometry of the representation changes: $\varphi \neq \varphi'$

MNIST dataset (2D representation space)



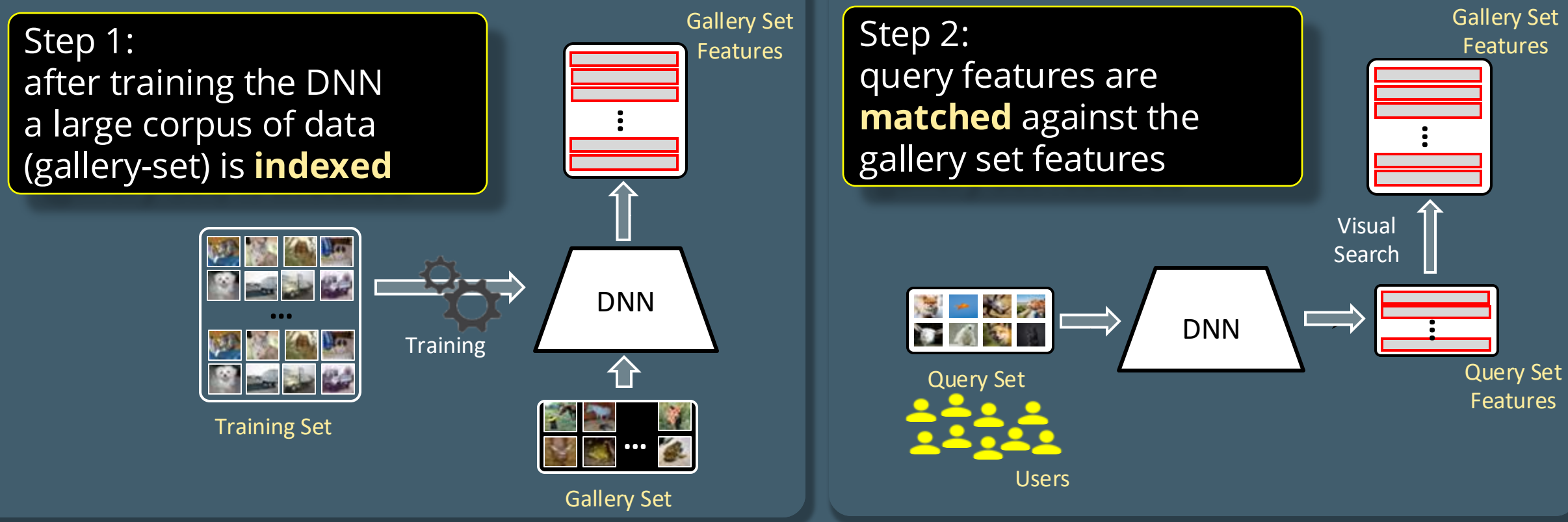
COMPATIBLE LEARNING

THE CONTEXT

Retrieval Systems

Visual search systems address the problem of finding similar data in a gallery-set

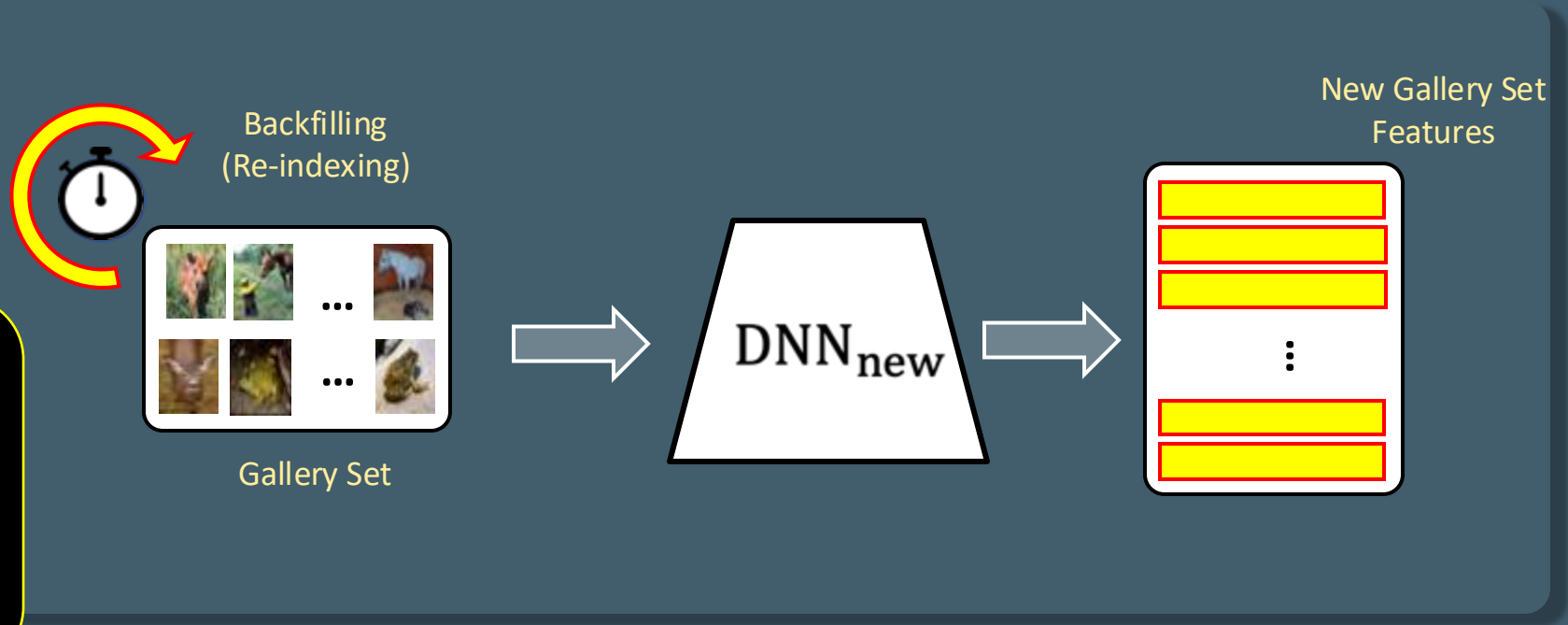
The core part includes an
online service of the embedding model and a large-scale vector database



Backfilling 👎

For traditional model upgrades the old model is not replaced by the new one until the embeddings of all the images in the database are **re-computed by the new model**.....

Which takes days or weeks for a large amount of data



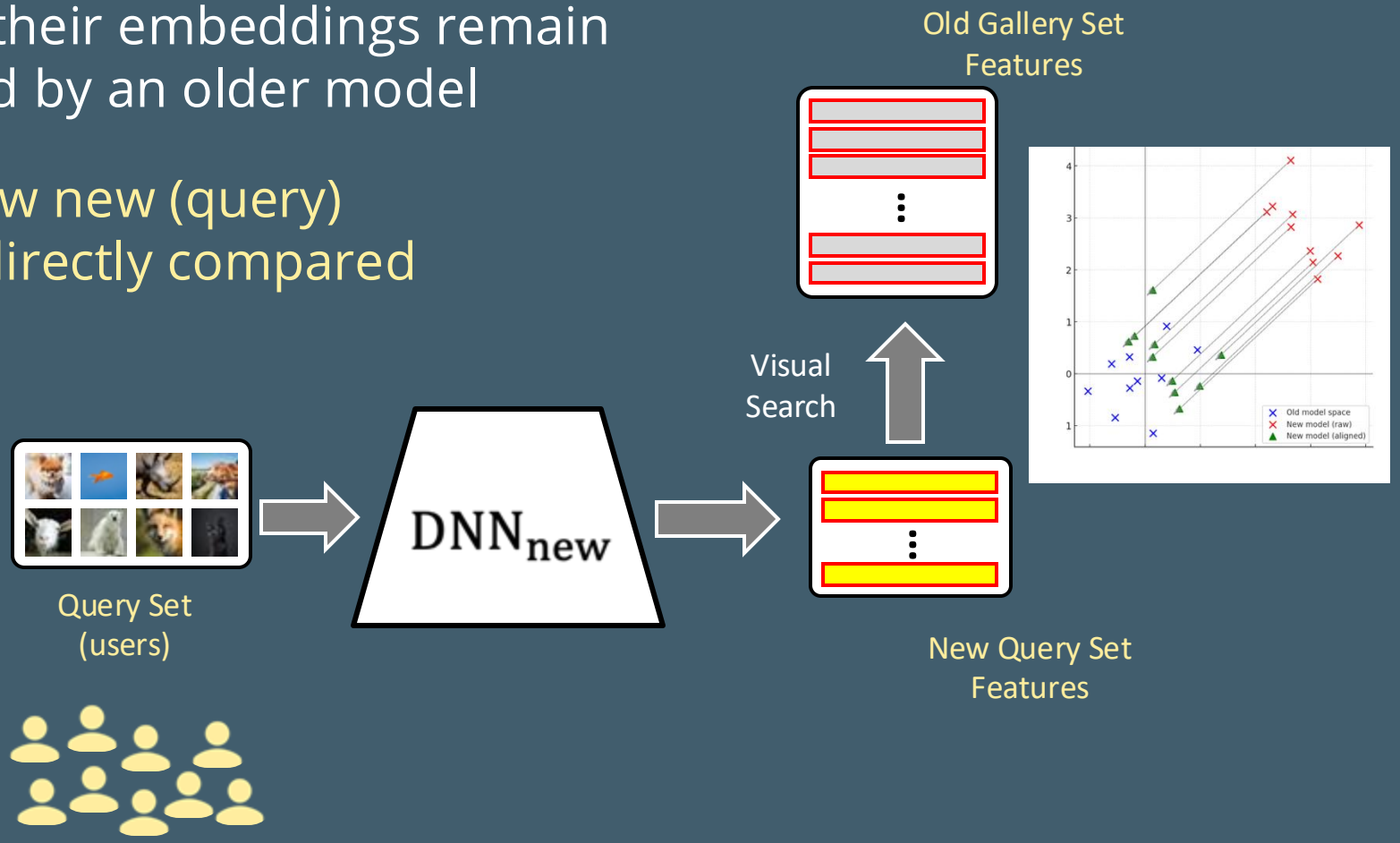
Issues:

- ✓ Time consuming
- ✓ Carbon Footprint
- ✓ Storage
- ✓ Privacy

Compatibility Learning 👍

Compatibility learning is learning embedding alignment, i.e. training new models so that their embeddings remain comparable with those produced by an older model

Compatible representations allow new (query) and old (gallery) features to be directly compared

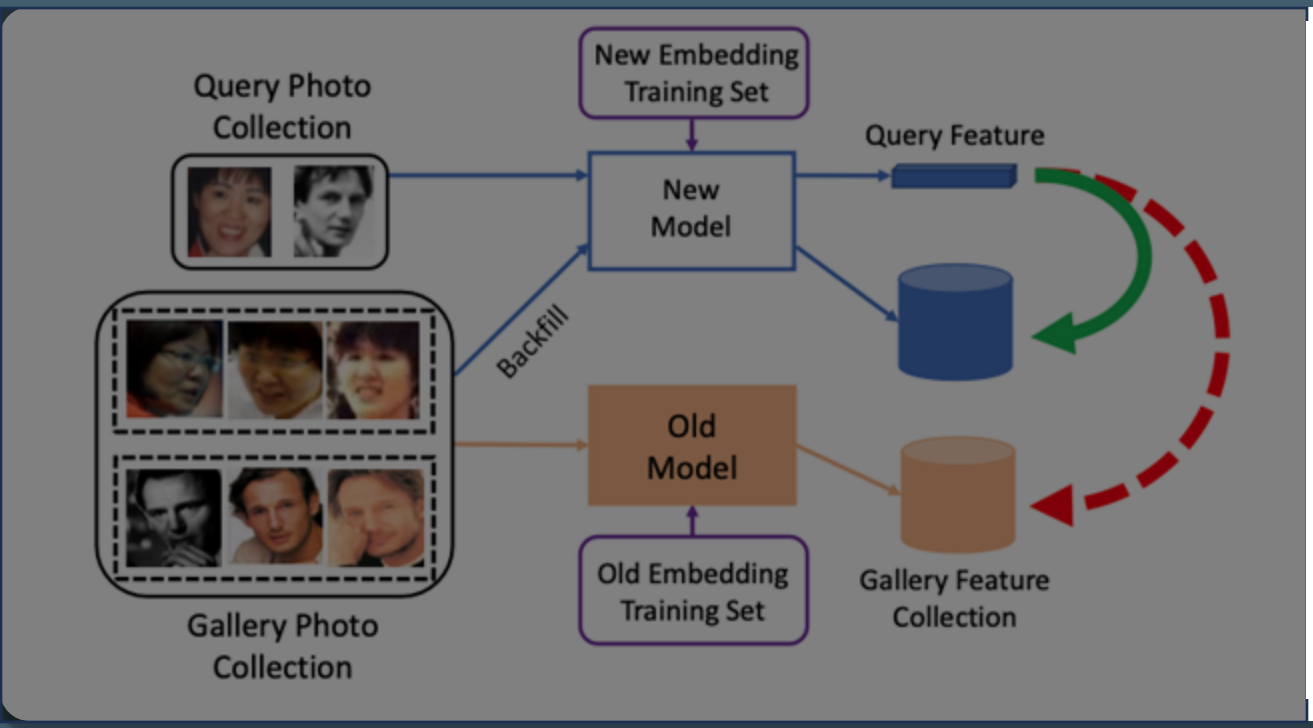


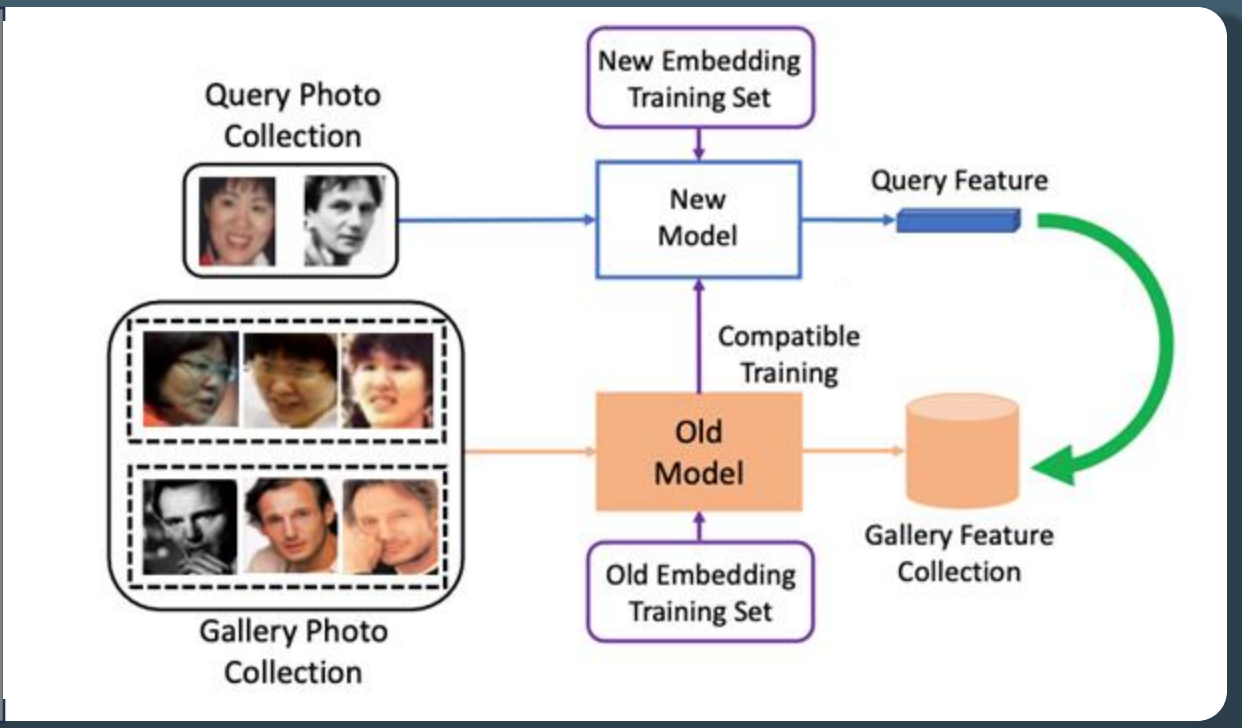
Benefits:

- ✓ Avoid re-indexing gallery-set
- ✓ Privacy-preserving
- ✓ Memory-efficient

Backfilling vs Compatible Learning

With a backward compatible representation
direct comparison becomes possible



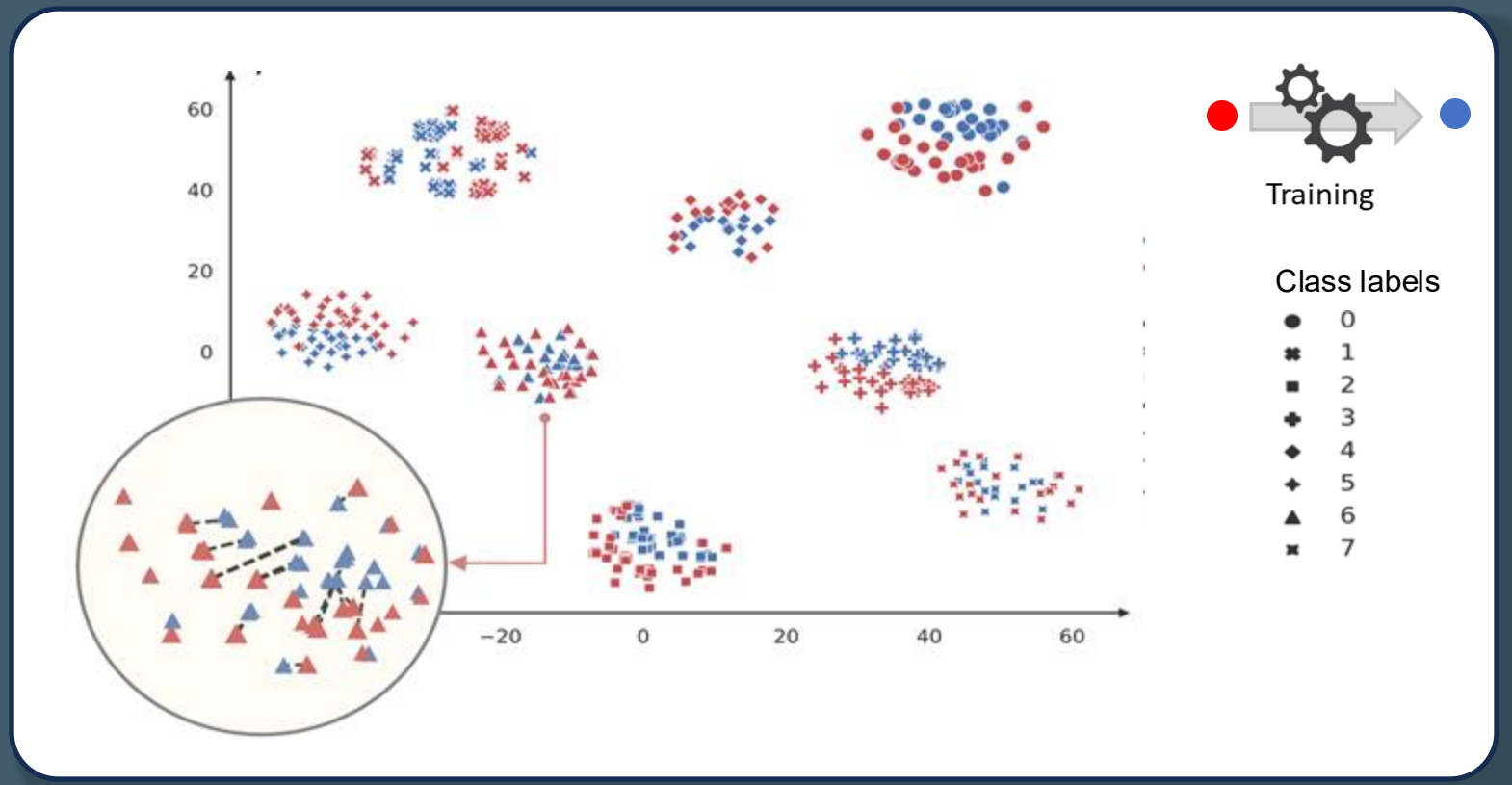


[*] Y. Shen et al., CVPR 2020

pictures
authored by Y. Shen

Compatible Learning Main Goal

We are interested in learning a new representation that **does not change** its geometrical structure after updating, i.e. that **is aligned** to the old one

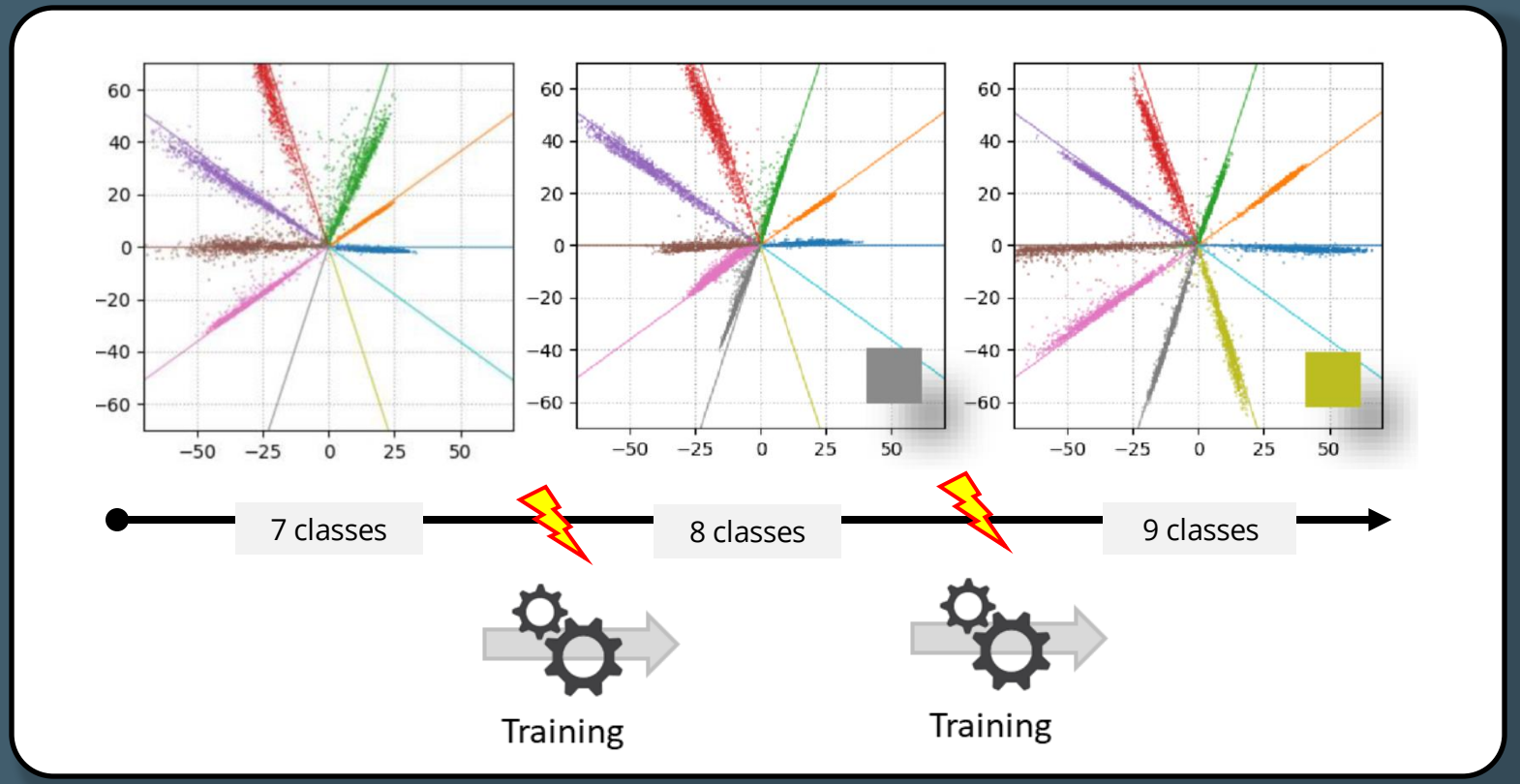


Same classes

picture
authored by Meng

Compatible Learning Main Goal

We are interested in learning a new representation that **does not change** its geometrical structure after updating, i.e. that **is aligned** to the old one



Adding new classes

MNIST dataset (2D representation space)

Backward Compatibility [*]

Backward Compatibility Criterion:

$$\begin{aligned}
&\operatorname{dist}\left(\phi_{\text{new}}(\mathbf{x}_i), \phi_{\text{old}}(\mathbf{x}_j)\right) \leq \operatorname{dist}\left(\phi_{\text{old}}(\mathbf{x}_i), \phi_{\text{old}}(\mathbf{x}_j)\right) \\
&\qquad\qquad \forall\,(i,j) \in \left\{(i,j)\,|\,y_i = y_j\right\} \\
&\qquad\qquad \text{and} \\
&\operatorname{dist}\left(\phi_{\text{new}}(\mathbf{x}_i), \phi_{\text{old}}(\mathbf{x}_j)\right) \geq \operatorname{dist}\left(\phi_{\text{old}}(\mathbf{x}_i), \phi_{\text{old}}(\mathbf{x}_j)\right) \\
&\qquad\qquad \forall\,(i,j) \in \left\{(i,j)\,|\,y_i \neq y_j\right\},
\end{aligned}$$

for all pairs of samples $\mathbf{x}_i$ *and* $\mathbf{x}_j$

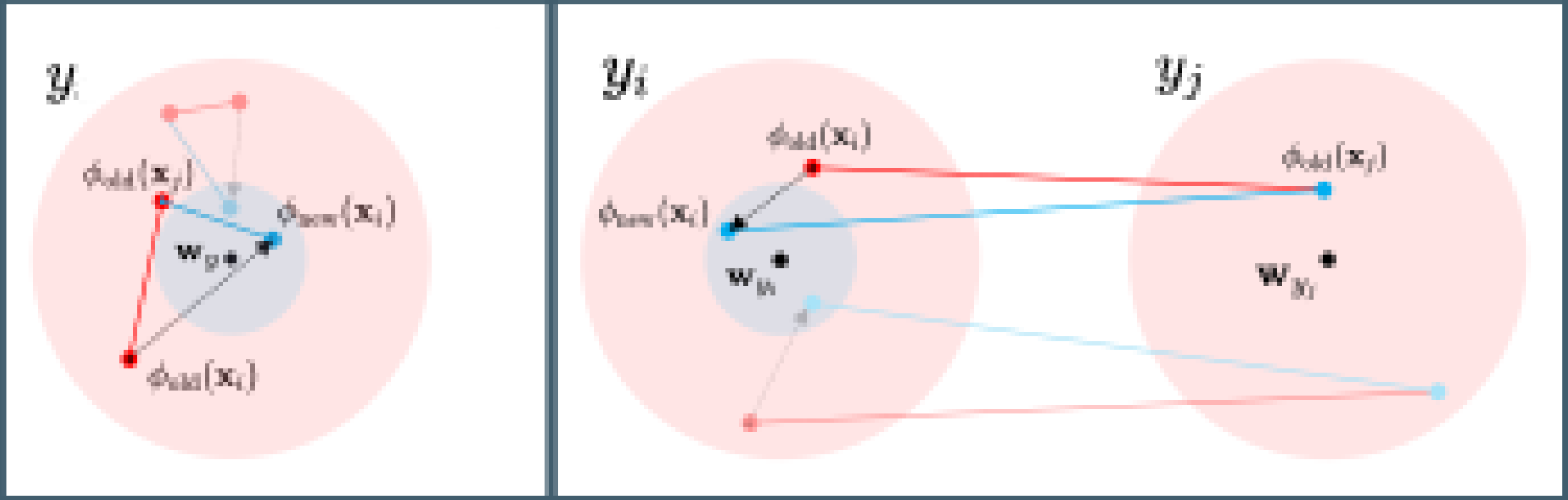
The new embedding when used to compare against the old embedding must be at least as good as the old one in separating images from different classes and grouping those from the same classes [*]

[*] Y. Shen et al., CVPR 2020

Backward Compatibility

Backward compatibility must hold for each pair of feature instances

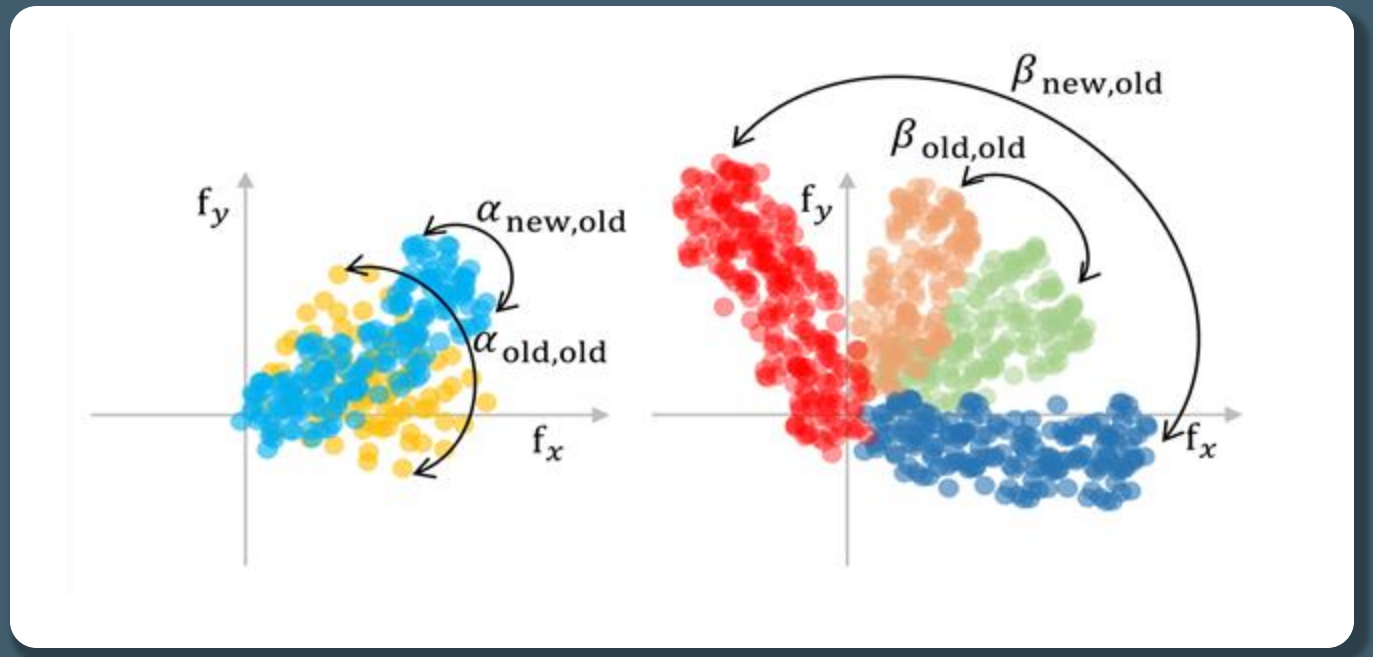
The Backward Compatibility criterion is **intractable** at large scale and in the open-set setting. It entails testing the gallery exhaustively
The constraints could not hold for a few feature instances



Backward Compatibility

The new model must be at least as good as the old one in:

- ✓ **grouping** images from the same classes
  $(\alpha_{\text{new,old}} \leq \alpha_{\text{old,old}})$
- ✓ **separating** images from different classes
  $(\beta_{\text{new,old}} \geq \beta_{\text{old,old}})$

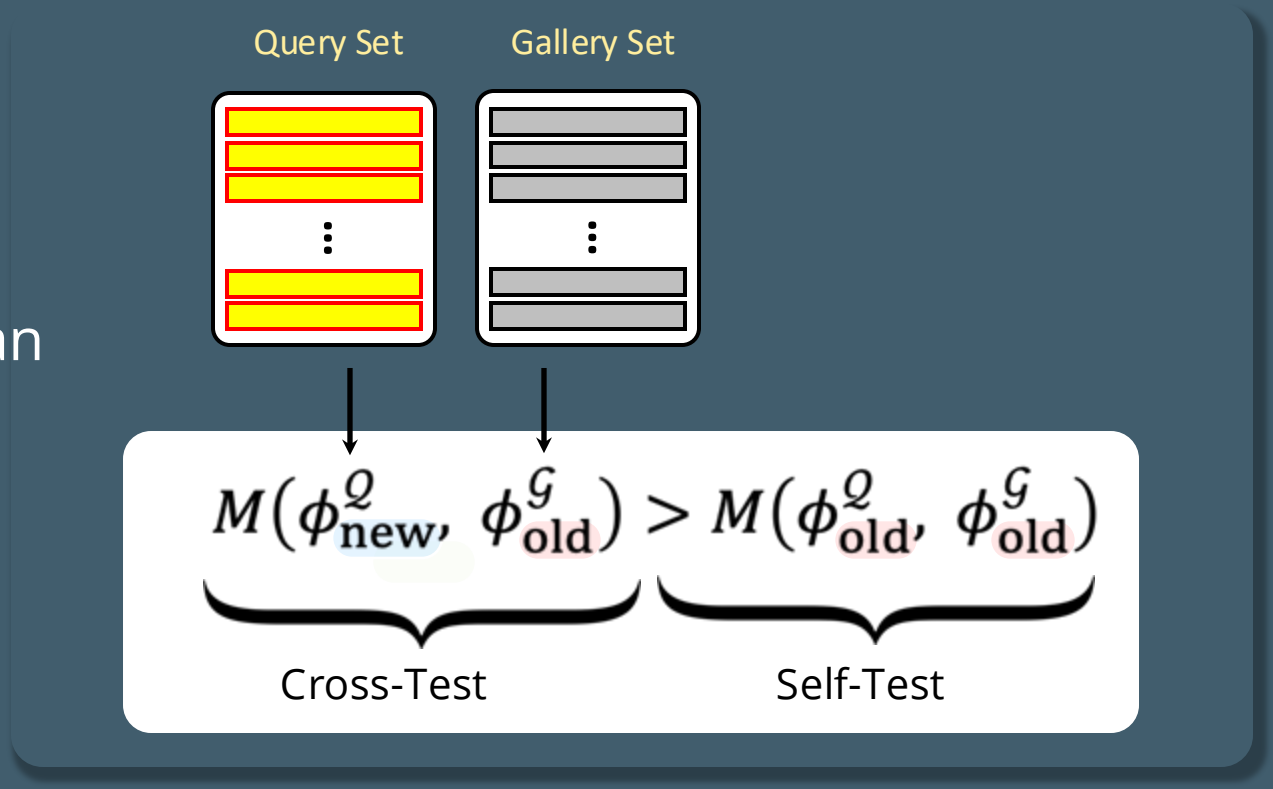


Empirical Compatibility [*]

Empirical Compatibility criterion: backward compatibility is achieved when the accuracy using φ_{new} for queries without backfilling gallery images surpasses that of using φ_{old}

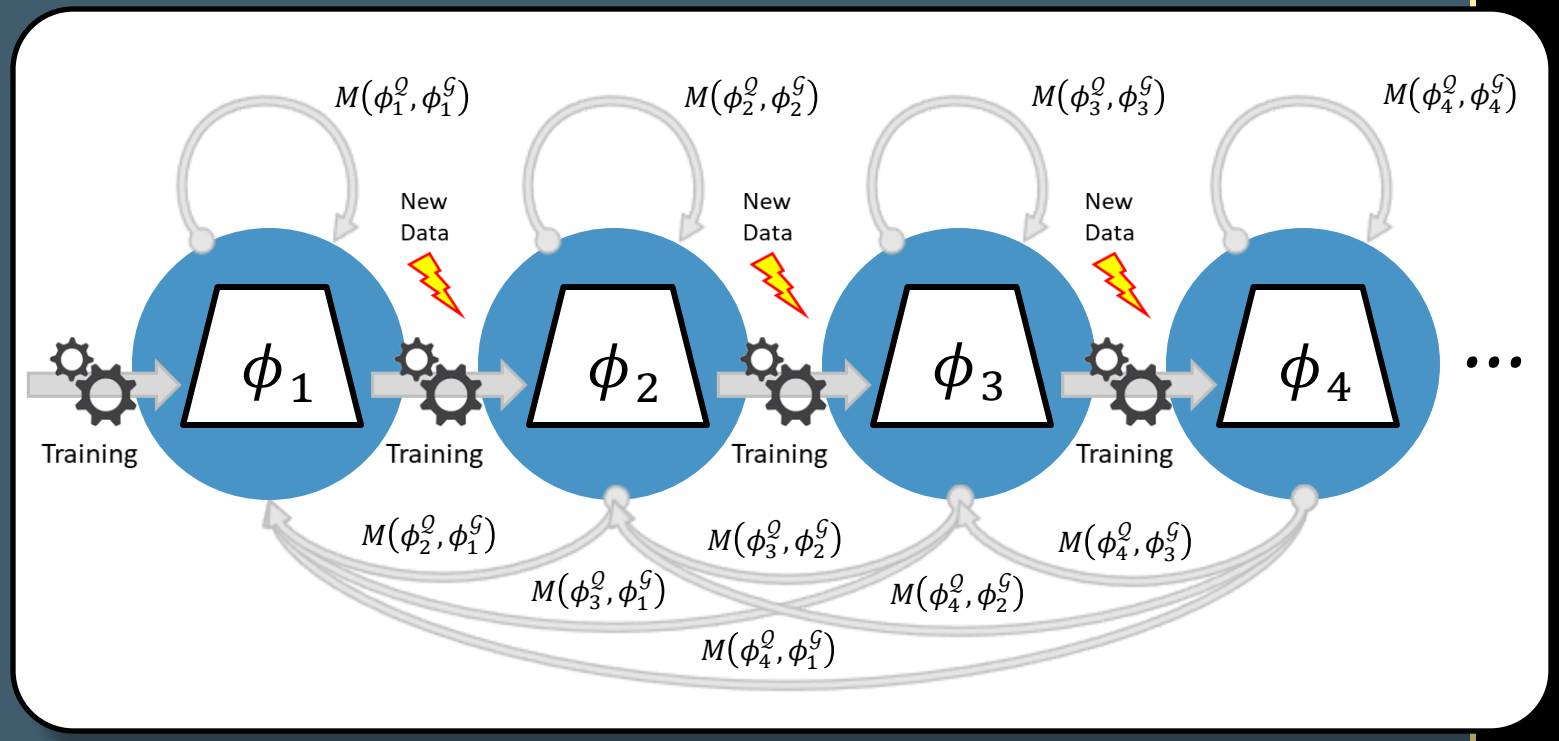
In other words:

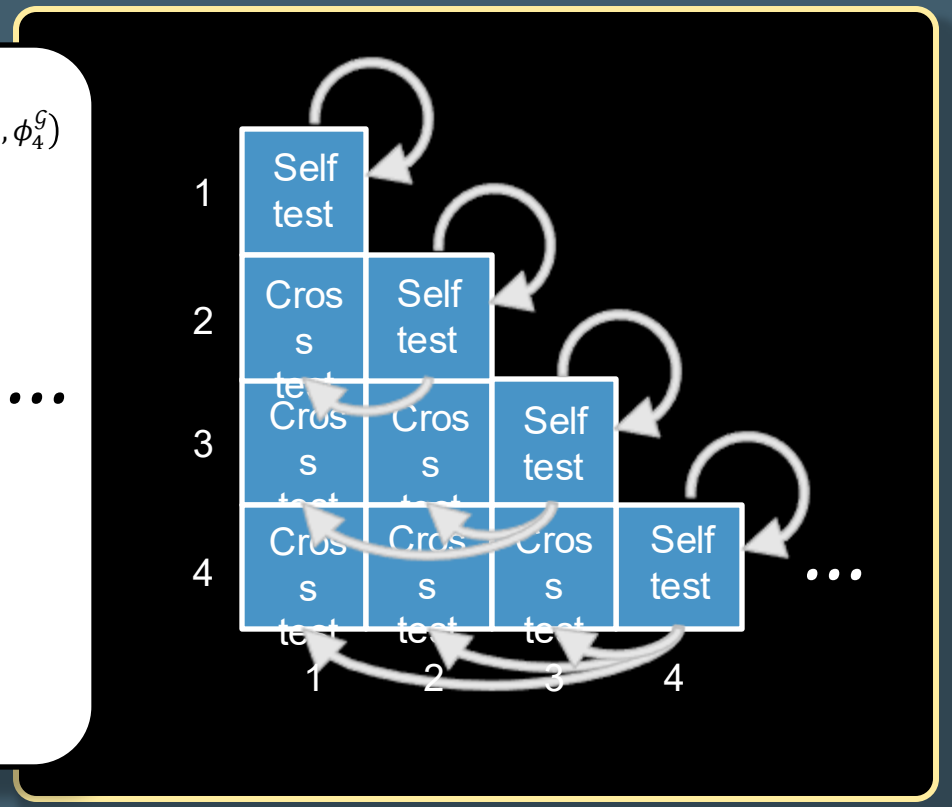
compatibility between two models is achieved if the cross-test is higher than the self-test of the older model



[*] Y. Shen et al., CVPR 2020

Multi-step Compatibility Matrix [*]

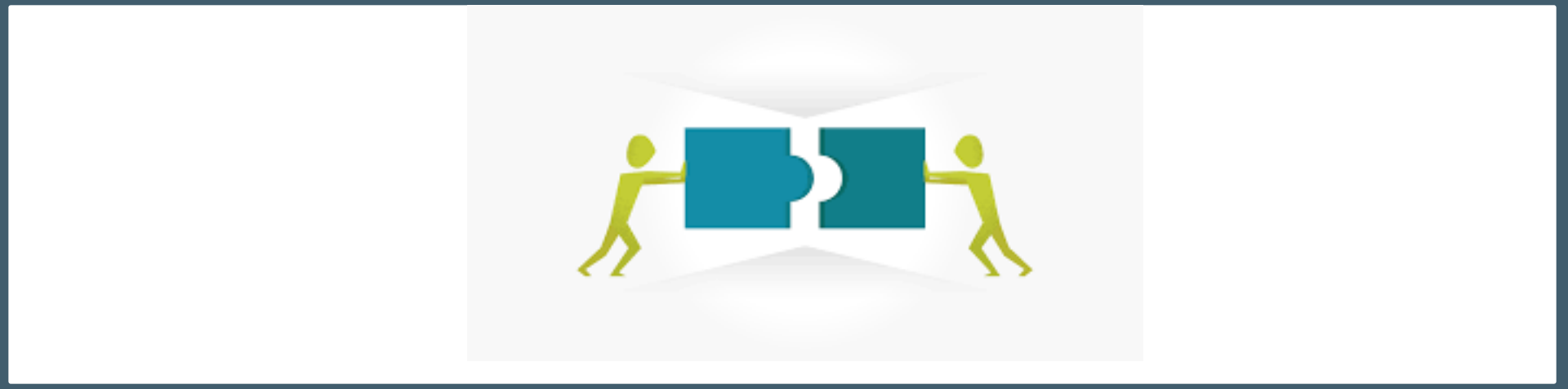




Compatibility between two models is achieved if the
cross-test is higher than the self-test of the older model

[*] N. Biondi et al. , IEEE TPAMI 2023

COMPATIBLE LEARNING



SOLUTIONS

Proposals for Solutions

Using Regularization:

aims at upgrading models with additional loss functions that impose constrains with respect to the previous model

- ✓ BCT (CVPR2020)
- ✓ UniBCT (IJCAI2022)
- ✓ AdvBCT (CVPR2023) …

Using Mapping

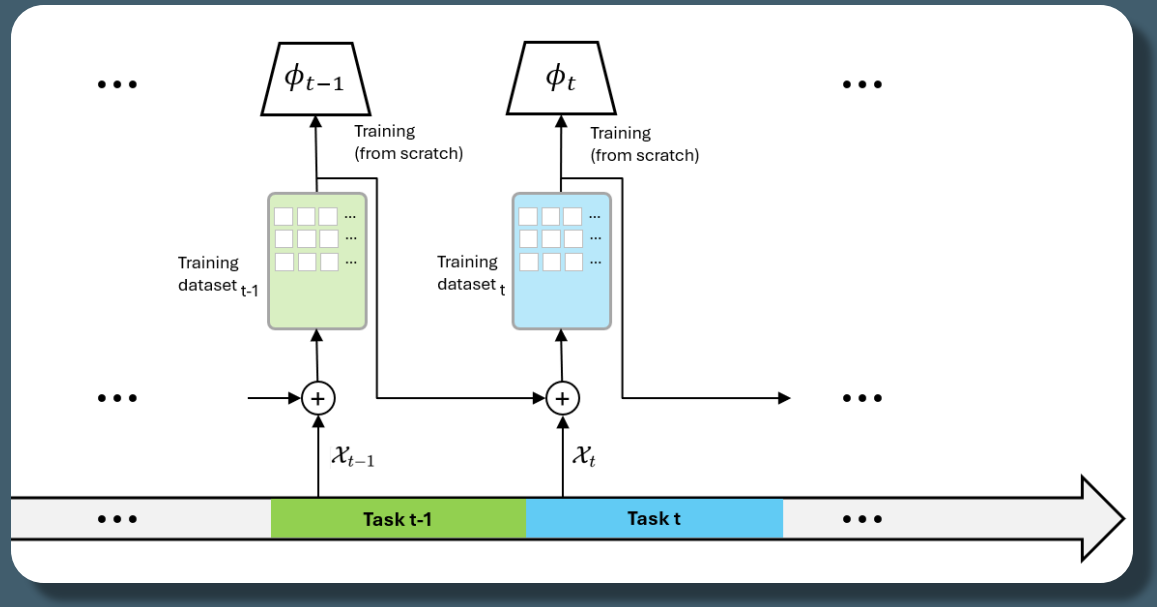
finds compatible mappings between the previous and upgraded models, where the upgraded models already exist

- ✓ CMC-RBT (BMVC2020)
- ✓ LCE (ICCV2021)
- ✓ FCT (CVPR2022) …

Using Architectural Changes:

impose constrains on the network architecture that force feature alignment

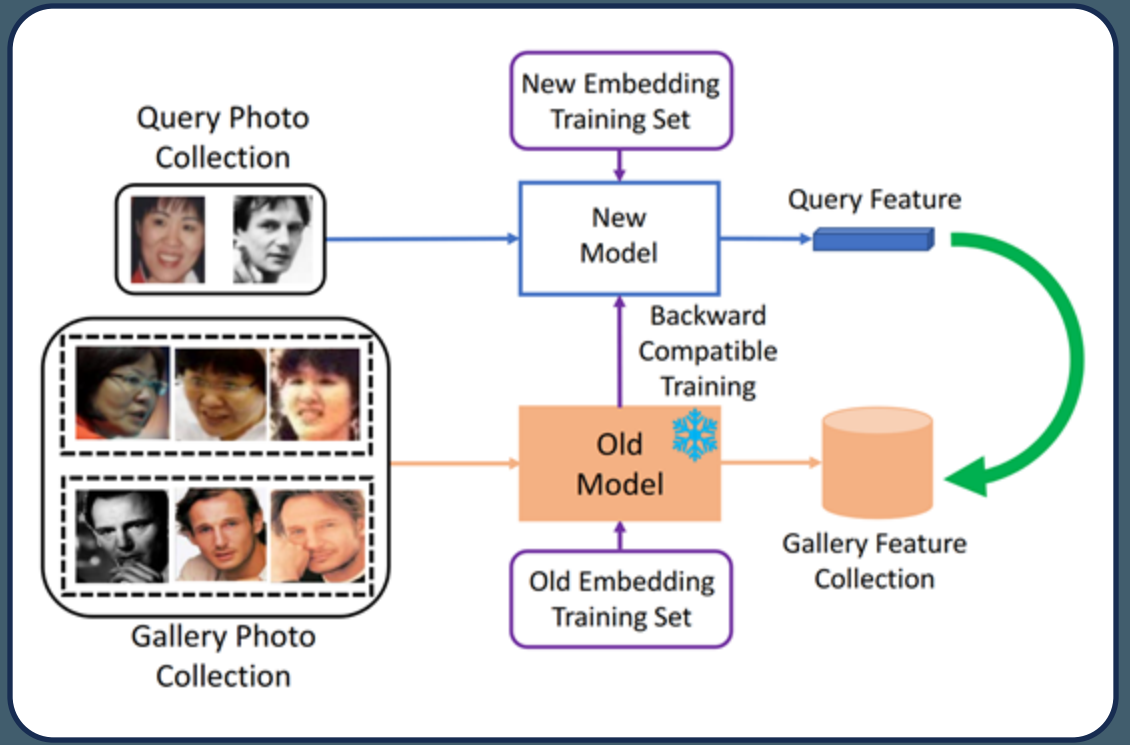
- ✓ CoReS (TPAMI2023)
- ✓ BT^2 (ICCV2023)
- ✓ OCA (ECCV2024)



Backward Compatible Training (BCT) [*]

BCT learns the new compatible model by **freezing the old classifier** and using it as a regularizer to align the new class prototypes to the old ones

The new model learns from both the new and the old data



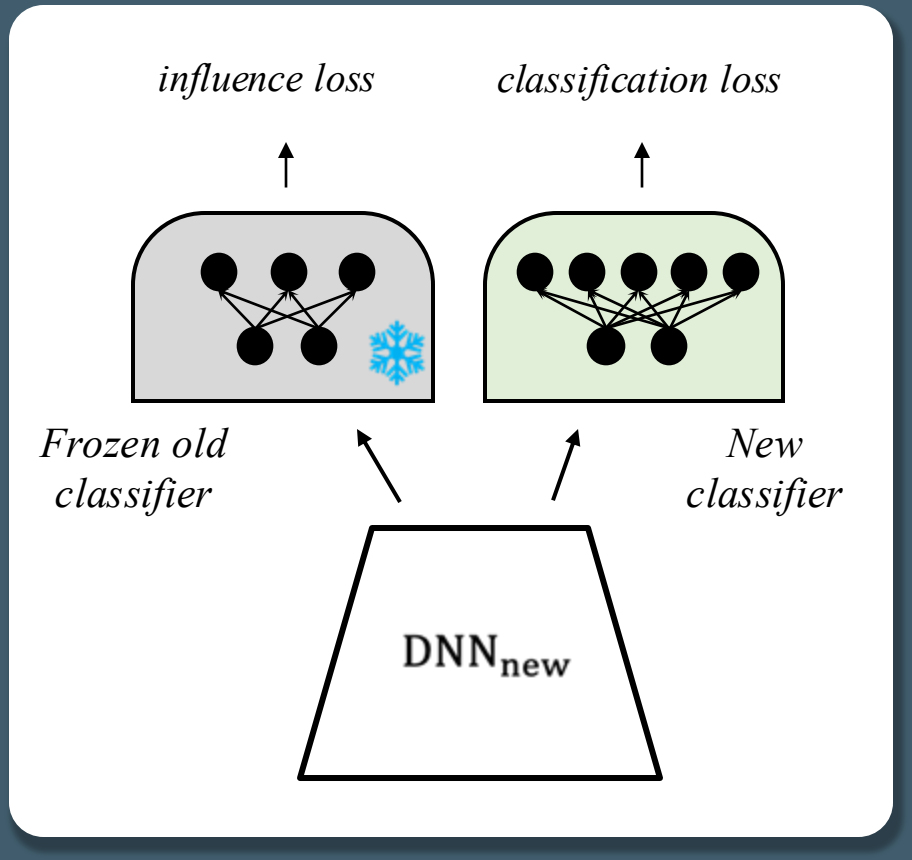
picture
authored by Y. Shen

[*] Y. Shen et al., CVPR 2020

Adds **influence loss** to the **classification loss** to bias the solution towards the old classifier

$$L_{\mathrm{BCT}}(w_c, w_\phi; \mathcal{T}_{\mathrm{new}}, \mathcal{T}_{\mathrm{BCT}}) = L(w_c, w_\phi; \mathcal{T}_{\mathrm{new}}) + \\ + \lambda L(w_{c\ \mathrm{old}}, w_\phi; \mathcal{T}_{\mathrm{BCT}})$$

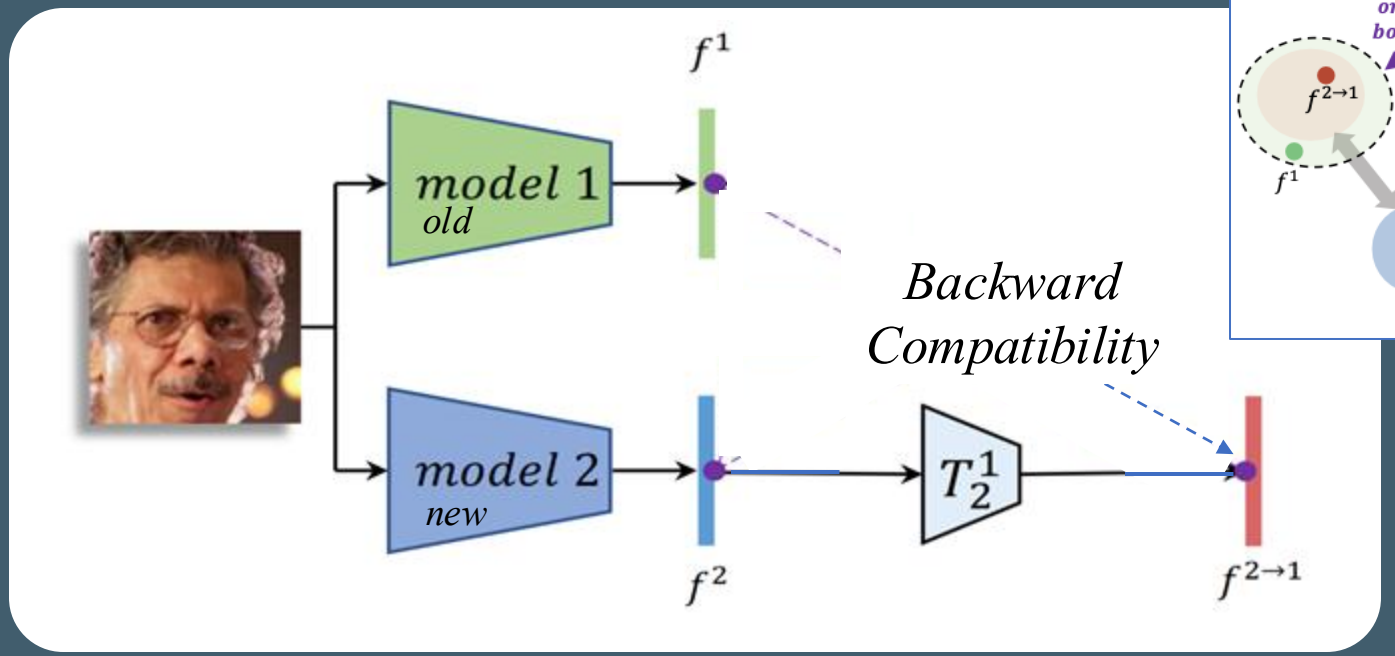
- ✓ $T_{BCT} = T_{old}$ computes the influence loss only on the old training data
- ✓ $T_{BCT} = T_{new}$ computes the influence loss on both the old and new training data. For new classes, *synthesized* classifier weights are created by computing the average feature vector of T_{old} on the images in each class

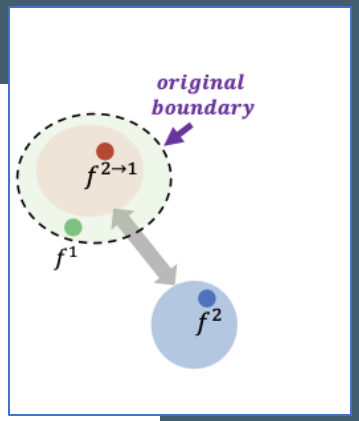


Learning Compatible Embeddings (LCE) [*]

Compatibility in LCE is achieved by **learning transformations** that map embeddings and class centers from one feature space to another

The method can work in backward / forward / direct manner





[*] Q. Meng et al., ICCV 2021

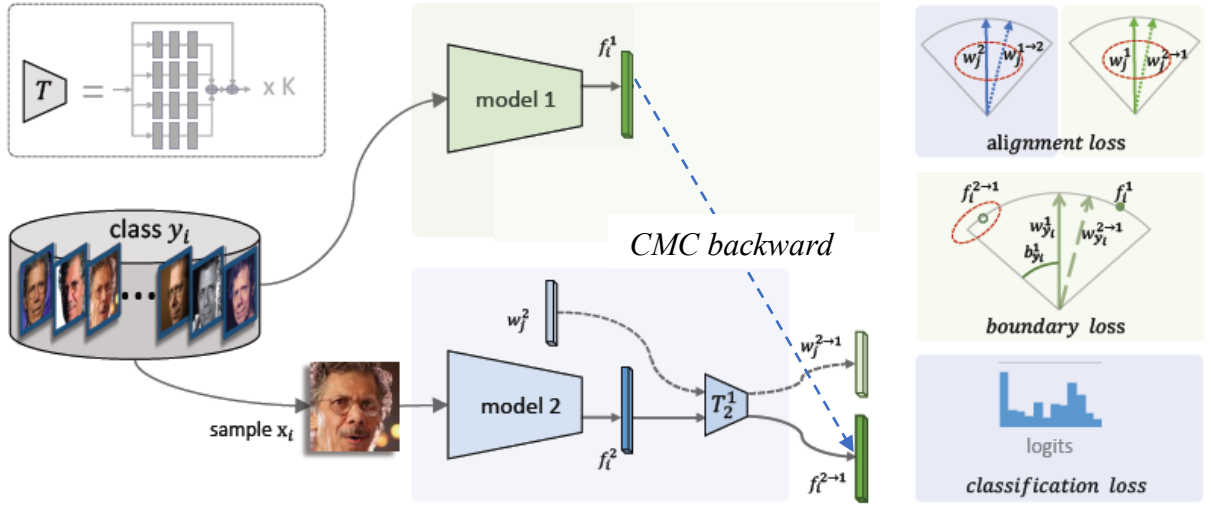
picture
authored by Q. Meng

Works in a point-to-set manner instead of employing point-wise constraints using:

- ✓ **Alignment loss** aligns class prototypes between two models
- ✓ **Boundary loss** enforces the mapped representation to have more compact intra-class distributions
- ✓ **Classification loss** to learn the data

$$L_{LCE} = \lambda_a L_a + \lambda_b L_b + L_c$$

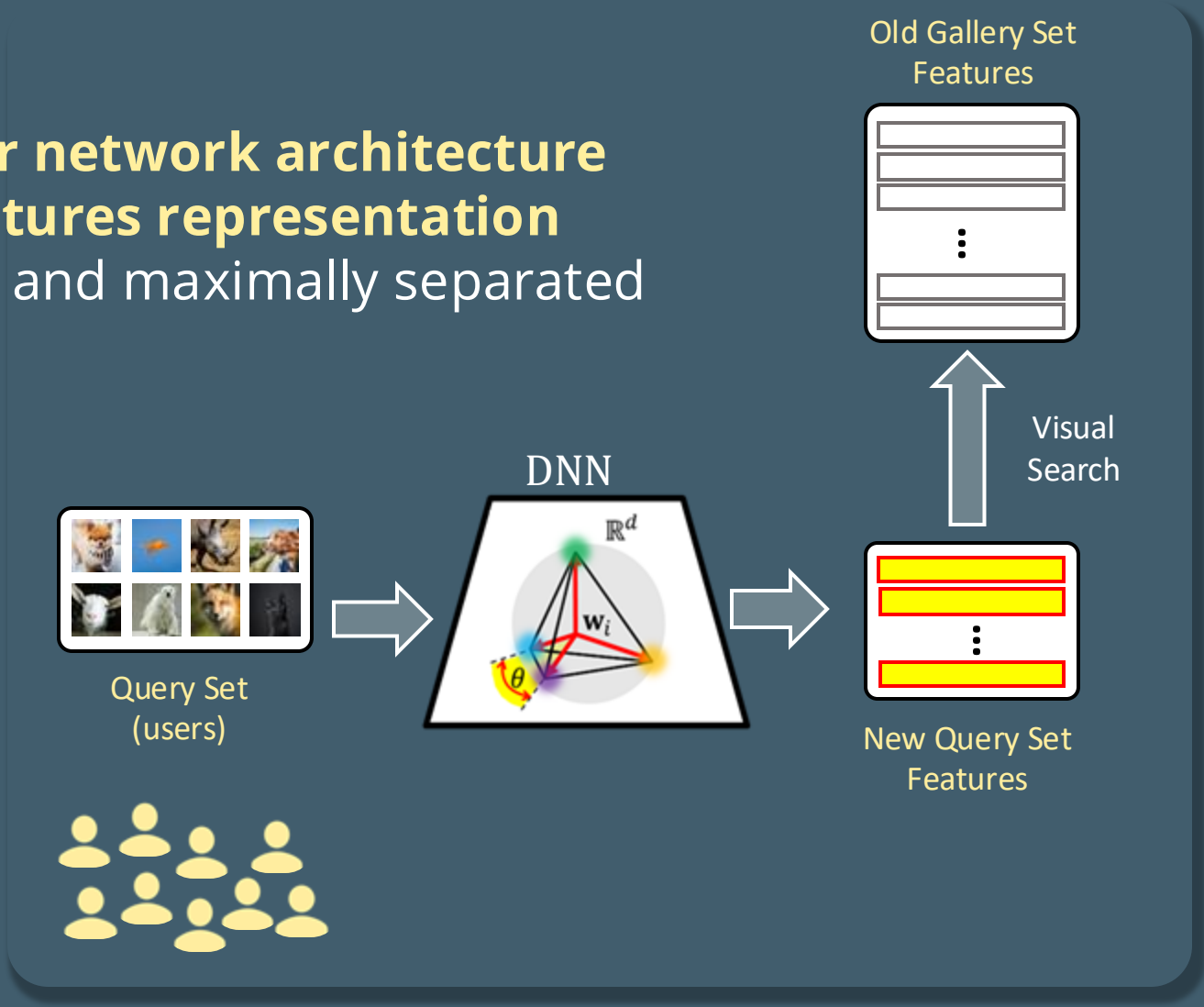
CMC mode main drawback:
with multiple updates LCE requires the **composition of multiple mapping functions**



picture
authored by Q. Meng

Compatible Representation via Stationarity (CoReS) [*]

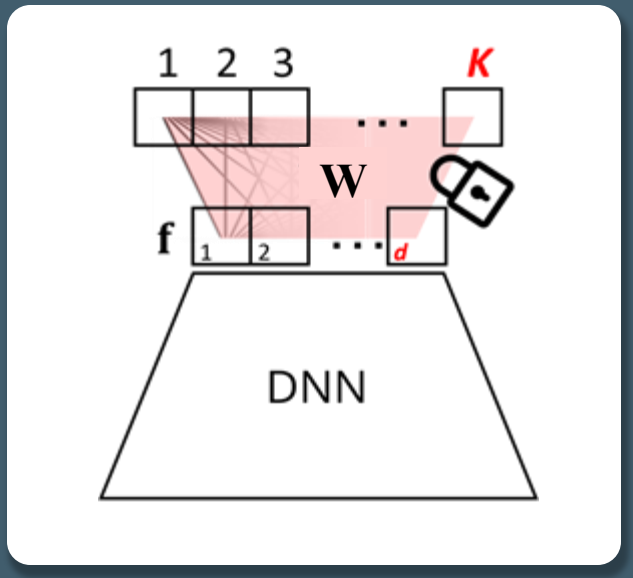
CoReS exploits the **Fixed Classifier network architecture with special configuration of features representation** to provide stationary embeddings and maximally separated representations



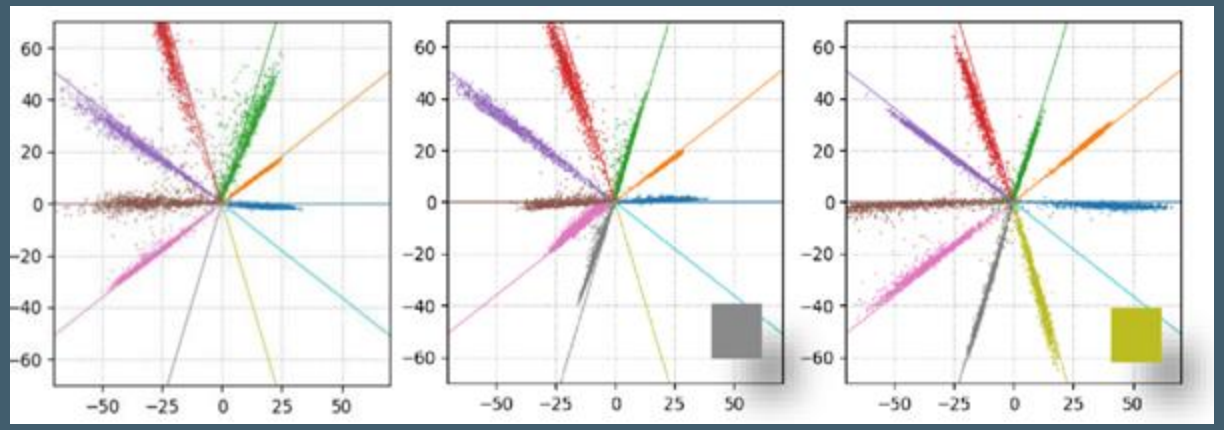
[*] N. Biondi et al., IEEE TPAMI 2023

CoReS Fixed classifiers

With **Fixed Classifiers** [*] the final layer of the CNN is **fixed** and the dynamic adjustment of the decision boundaries of class feature representations is demanded to the previous layers



Weights can be regarded **as fixed angular references** to which features align**.** This allows to define in advance where features will be projected, i.e. **enforce stationarity** over updatings

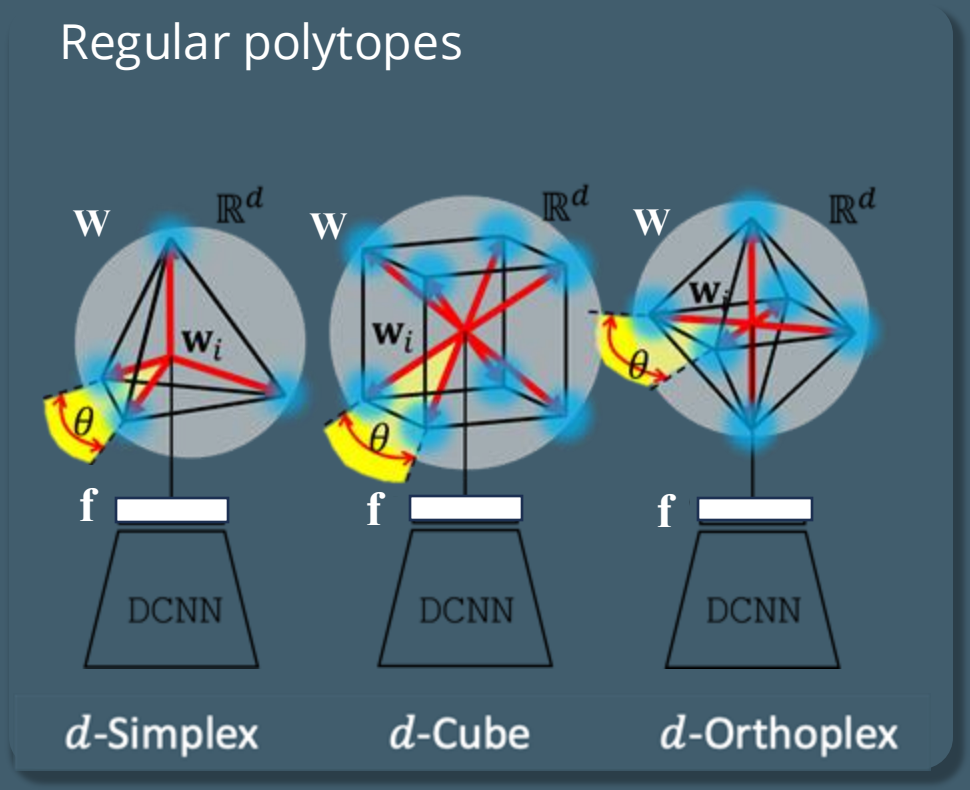


[*] E. Hoffer et al., ICLR 2018

CoReS Regular Polytopes

Maximally separated features are obtained by setting the weights of the Fixed Classifier to fixed values taken from the coordinate vertices of a **Regular Polytope**, following a highly symmetrical arrangement in the embedding space [*]

Dimension d	1	2	3	4	≥ 5
Number of Regular Polytopes	1	∞	5	6	**3**



[*] F. Pernici et al., IEEE TNNLS 2021

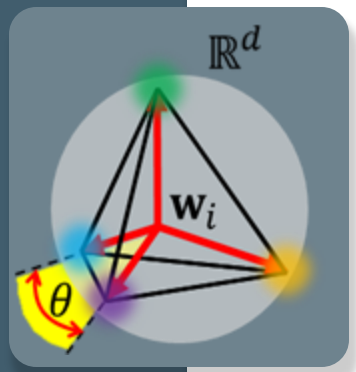
CoReS d-Simplex Fixed Classifier

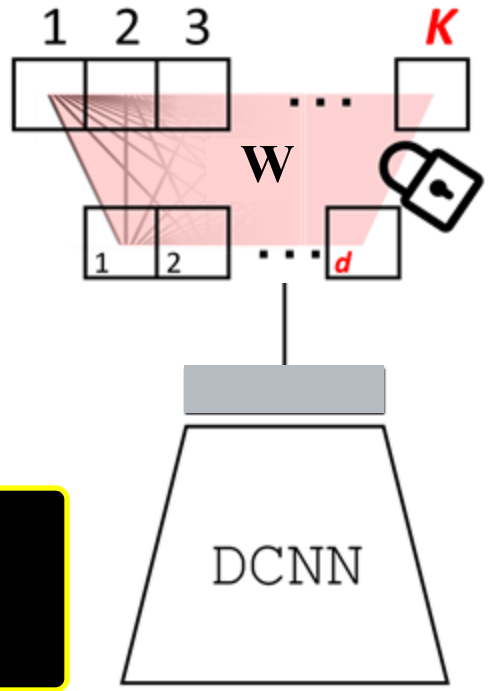
The d-**Simplex fixed classifier** of dimension d can accomodate a number of classes equal to the number of the d-Simplex vertices, i.e. $K = d + 1$

The d-Simplex weights

$$\mathbf{W} = \left[e_1, e_2, \ldots, e_{K-1}, \frac{1-\sqrt{K}}{K-1} \sum_{i=1}^{K-1} e_i \right]$$

e_i the standard basis of $\mathbb{R}^d$ with $i \in \{1, 2, \ldots, K-1\}$





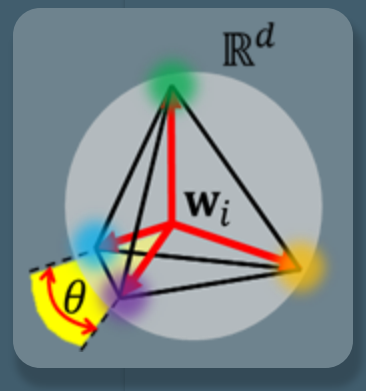
d-Simplex: same angle θ between classes

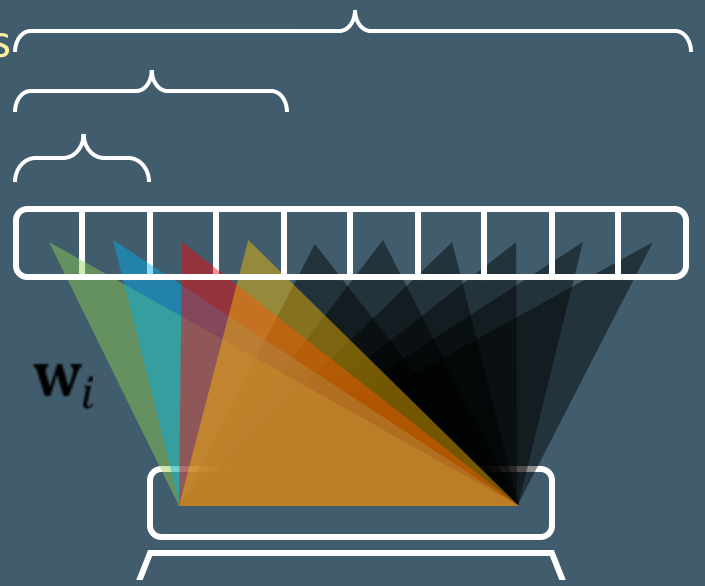
CoReS Class Pre-allocation

d-Simplex with **pre-allocated classes** allows stationarity when new classes are added with no changes to the spatial configuration of the features

No prior assumption on semantic similarity of future classes

$$\mathcal{L}_{\text{SCE}}(\phi_t) = -\sum_{B} \log \left(\frac{\exp\left(\mathbf{W}_{y_i}^{\top} \phi_t(\mathbf{x}_i)\right)}{\sum_{j=1}^{K_t} \exp\left(\mathbf{W}_j^{\top} \phi_t(\mathbf{x}_i)\right) + \sum_{j=K_t+1}^{K} \exp\left(\mathbf{W}_j^{\top} \phi_t(\mathbf{x}_i)\right)} \right)$$





d-Simplex Stationarity Implies Compatibility

Theorem 1 (Stationarity implies Compatibility). *Let* $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K$ *be the class prototypes of a* d*-Simplex fixed classifier. Consider* ϕ_k *and* ϕ_t *as representation models learned up to the* k*-th and* t*-th tasks, respectively, within this classifier. The numbers of classes learned by each model are denoted by* K_k *and* K_t*, where* $K_k < K_t < K$*. Under the assumption that class hyperspherical caps shrink after model updates, it follows that* ϕ_k *and* ϕ_t *satisfy, on average, the compatibility inequalities as in Def. 1.*

d-Simplex Implies Compatibility

d-Simplex classifier with K pre-allocated classes satisfies the compatibility inequalities in expectation across model updates [*]

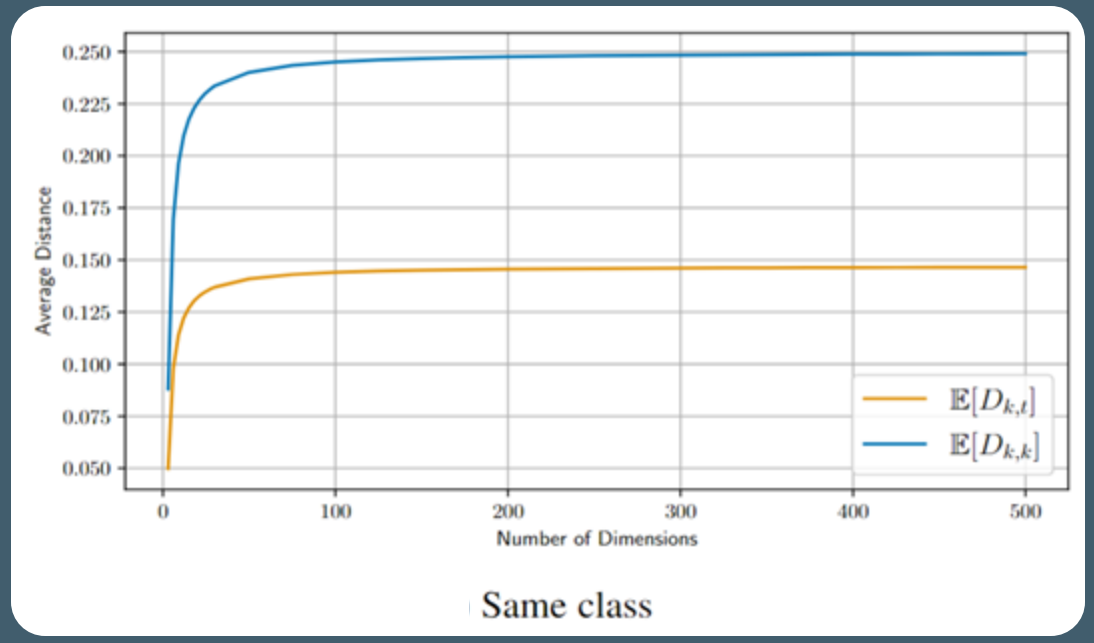
$$d(\phi_k(\mathbf{x}_i), \phi_t(\mathbf{x}_j)) \leq d(\phi_k(\mathbf{x}_i), \phi_k(\mathbf{x}_j))$$

$$\mathbb{E}[D_{k,t}] \leq \mathbb{E}[D_{k,k}]$$

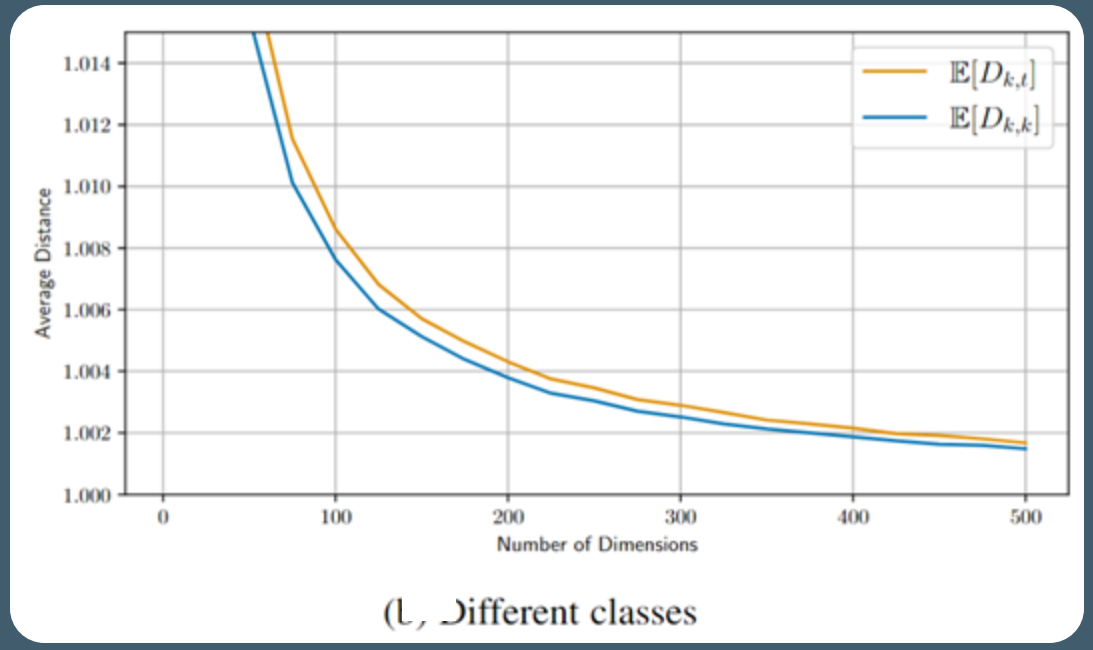
$$d(\phi_k(\mathbf{x}_i), \phi_t(\mathbf{x}_j)) \geq d(\phi_k(\mathbf{x}_i), \phi_k(\mathbf{x}_j))$$

$$\mathbb{E}[D_{k,t}] \geq \mathbb{E}[D_{k,k}]$$

A Montecarlo simulation



Same class



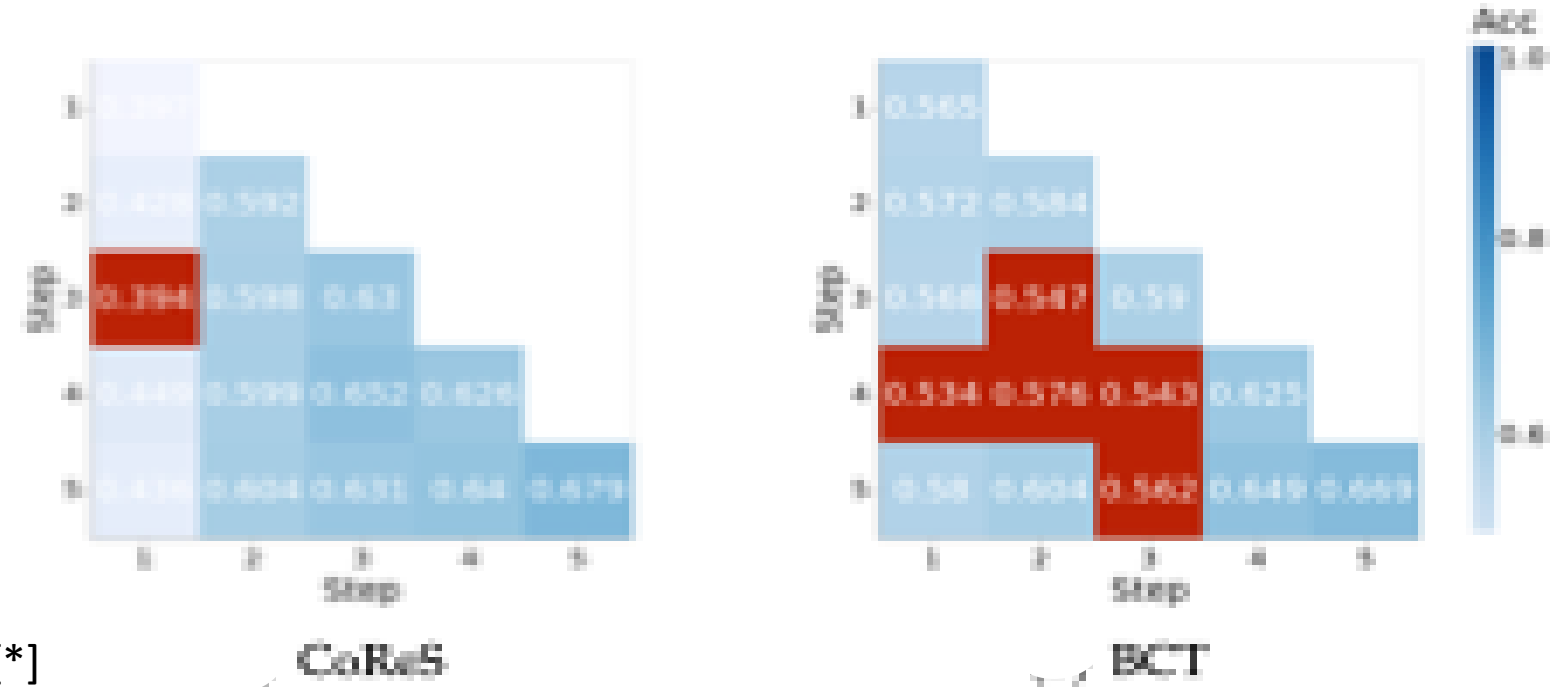
(b) Different classes

CoReS vs BCT

M = 1:1 Search Verification Accuracy

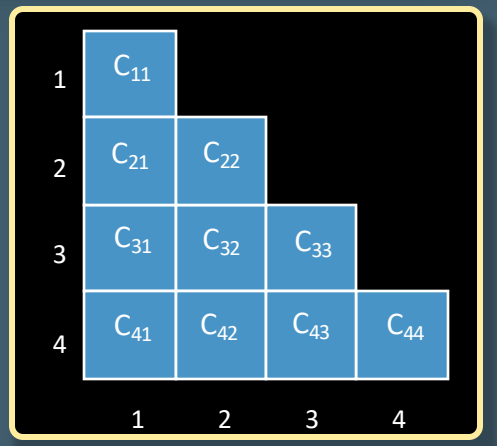
CIFAR100/10 Dataset

Open set



$$M\left(\phi^{\mathcal{Q}}_{\text{new}}, \phi^{\mathcal{G}}_{\text{old}}\right) > M\left(\phi^{\mathcal{Q}}_{\text{old}}, \phi^{\mathcal{G}}_{\text{old}}\right)$$

Cross-Test Self-Test



[*] N. Biondi et al., IEEE TPAMI 2023

Average Compatibility:
normalized count of the times Compatibility is assessed over T steps

$$AC = \frac{2}{T(T-1)} \sum_{t=2}^{T} \sum_{k=1}^{t-1} \mathbb{1}\left(C_{t,k} > C_{k,k}\right),$$

Average Accuracy:
average Accuracy assessed over T steps

$$AA = \frac{2}{T(T+1)} \sum_{t=1}^{T} \sum_{k=1}^{t} C_{t,k},$$

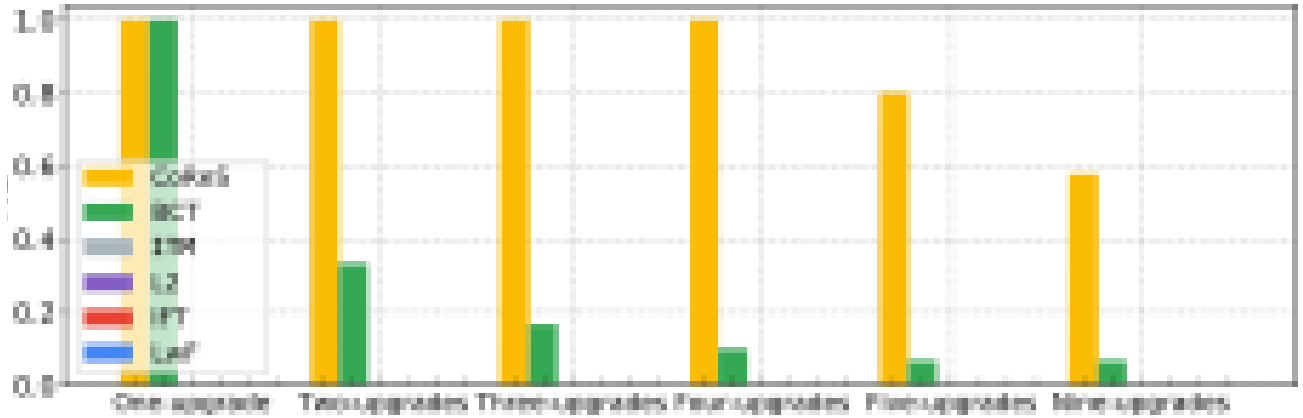
[*] N. Biondi et al., IEEE TPAMI 2023

M = 1:1 Search
Verification Accuracy

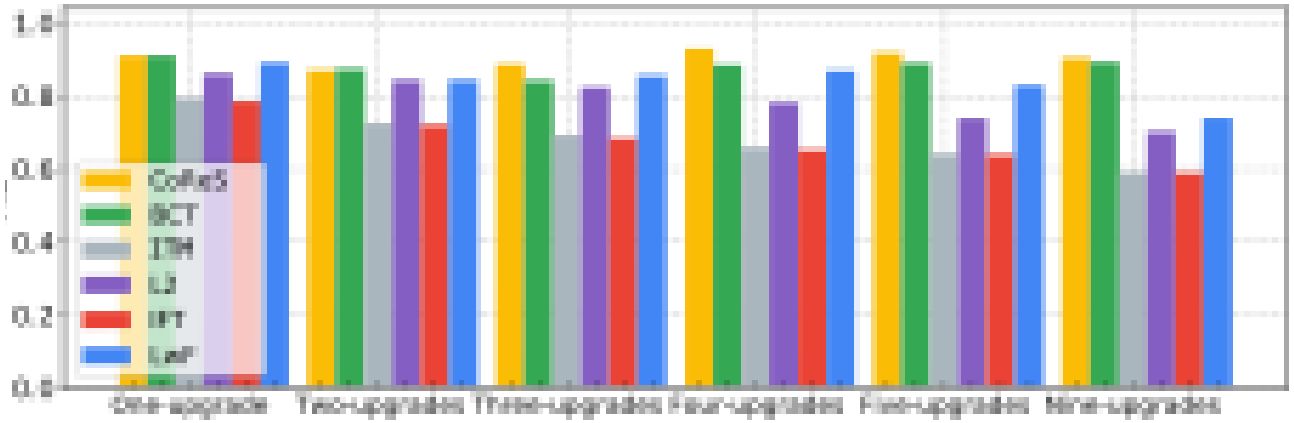
CASIA-WebFace/LFW Dataset

[*]

Average Compatibility



Average Accuracy

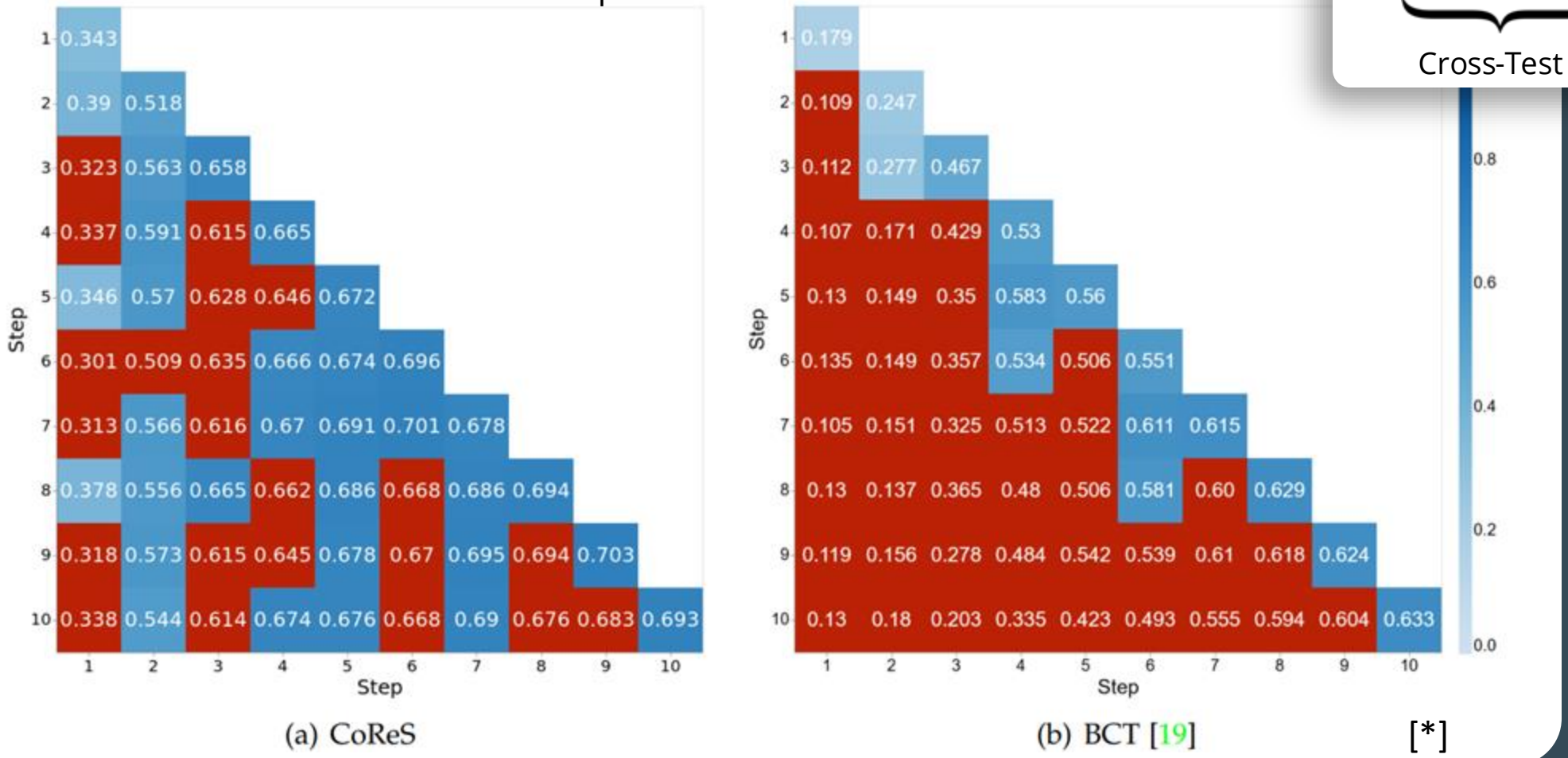


Comparative Evaluation CoReS vs BCT

CIFAR100/10 Dataset

$$M\underbrace{(\phi^{\mathcal{Q}}_{\text{new}}, \phi^{\mathcal{G}}_{\text{old}})}_{\text{Cross-Test}} > M\underbrace{(\phi^{\mathcal{Q}}_{\text{old}}, \phi^{\mathcal{G}}_{\text{old}})}_{\text{Self-Test}}$$

Open set



[*]

M = Verification Accuracy

[*] N. Biondi et al., IEEE TPAMI 2023

Drawbacks

Regularization mode main drawback:

- ✓ Models are trained with an explicit dependency from each other
- ✓ There is a performance loss due to the regularization constraints

Mapping mode main drawback:

- ✓ Models are trained independently from each other
- ✓ There is a performance loss due to the composition of multiple mapping functions

Architectural modification mode main drawback:

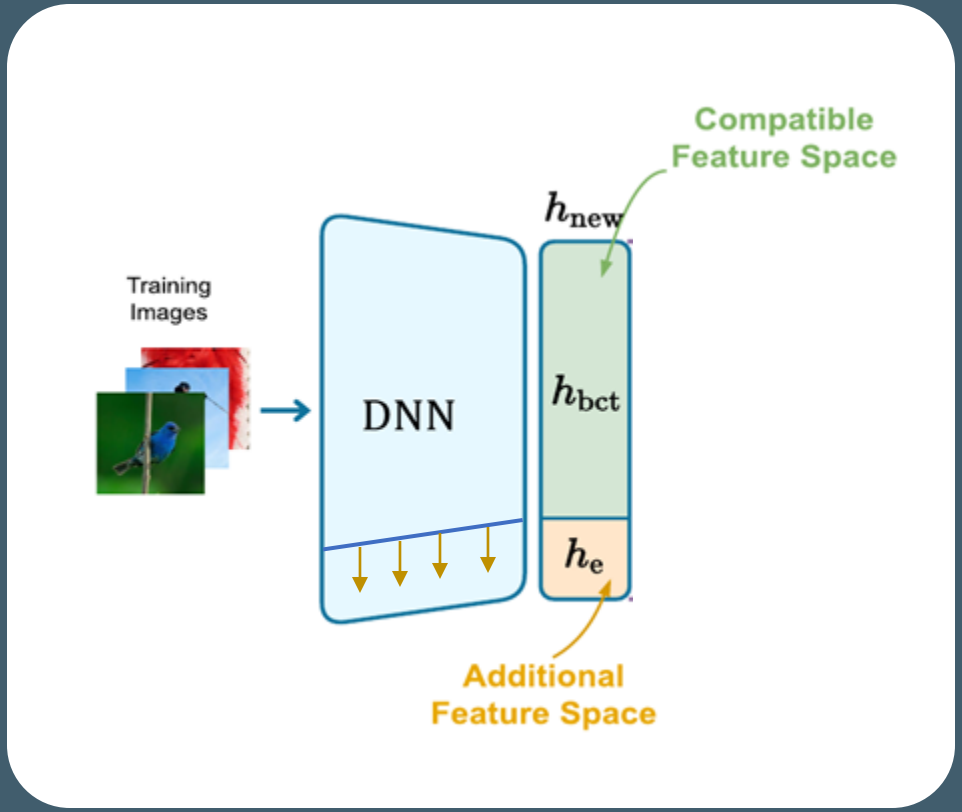
- ✓ Models are trained independently from each other
- ✓ The fixed architecture poses limits for their applicability in existing systems

Architectural changes: feature space expansion

BCT does not match the performance of a new independently trained model.

Feature space expansion strategies are used to reach higher retrieval performance.

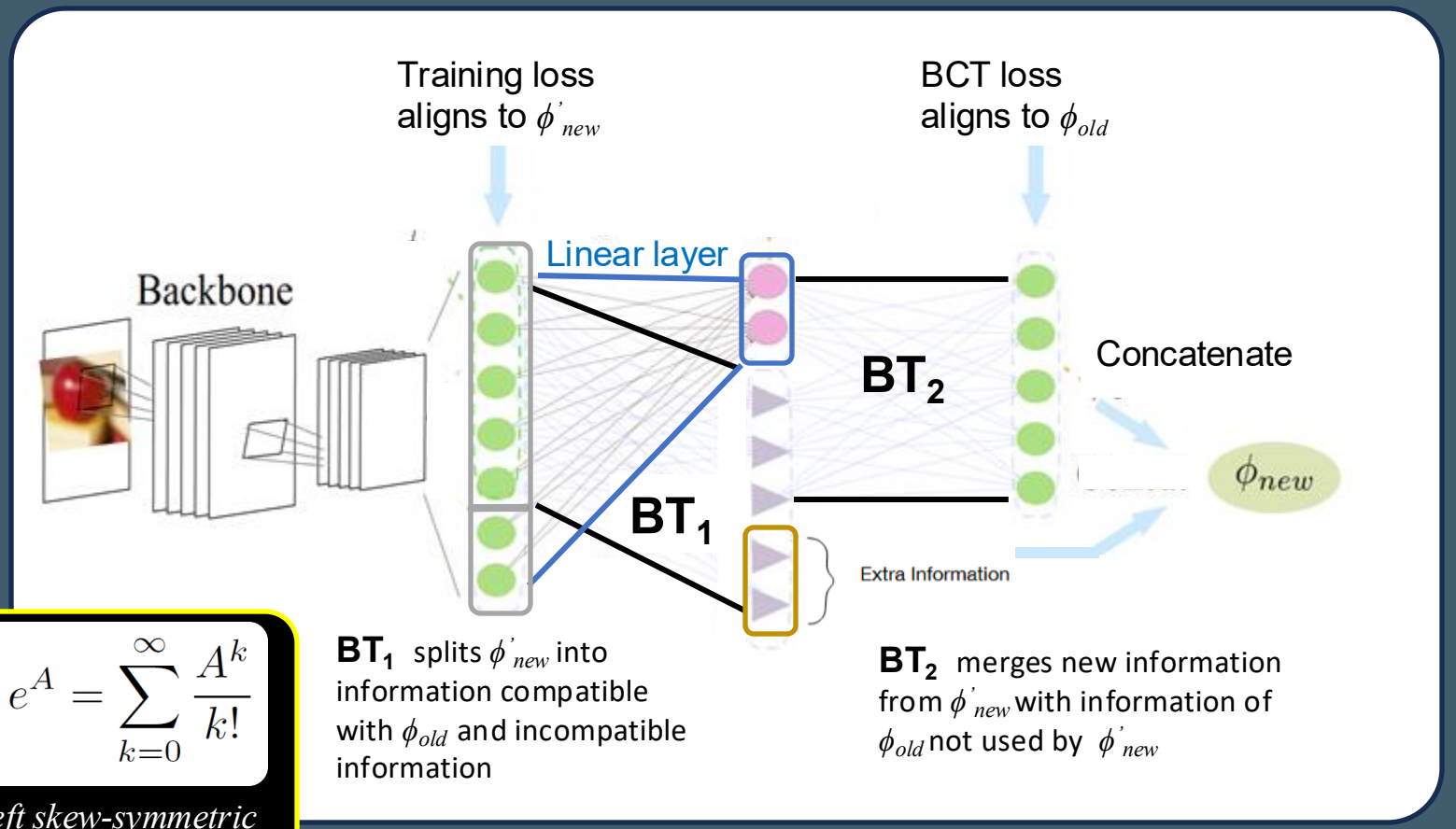
Backward compatibility is achieved by **adding extra-dimensions** and using **specific transformations.**



Backward-compatible Training with Basis Transformation (BT^2) [*]

Two basis transformations are exploited to automatically **pick out the information from ϕ'_{new} that is non compatible** with ϕ_{old}

This extra information is encoded on the **dimension orthogonal** to ϕ_{old}



$$\mathrm{BT} = e^{A} = \sum_{k=0}^{\infty} \frac{A^k}{k!}$$

A is a left skew-symmetric matrix

[*] Y. Zhou et al. ICCV 2023

Picture authored by Y. Zhou (modified)

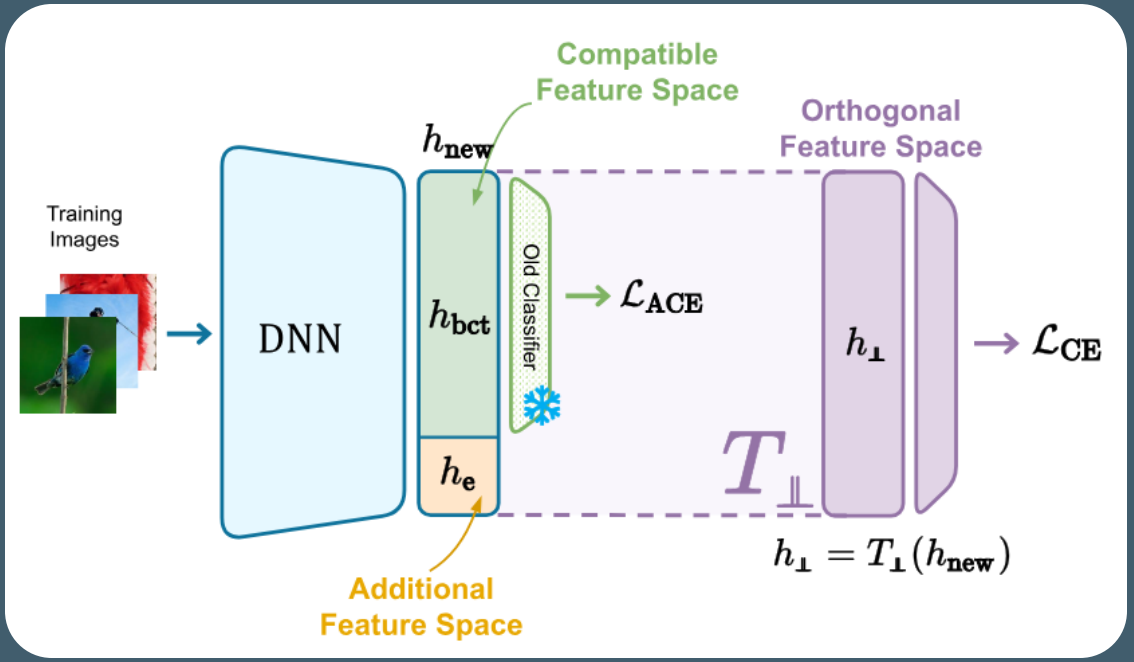
Backward-Compatible Aligned Representations via an Orthogonal Transformation Layer (OCA)[*]

OCA expands the feature space of the model and implements BCT on the **old** feature space.

The new embedding space (**old**+**extra**) is transformed by an **orthogonal transformation T**

T constraints the optimization process to avoid disruption of the compatible space by preserving the space geometry.

A φ'_{new} is not needed during training!



$$\mathcal{L}_{ACE} = \lambda_1 \cdot \mathcal{L}_{CE}(W_{old}, \phi_{new}(\mathbf{x})) + \lambda_2 \cdot \mathcal{L}_{\angle}$$

$$\mathcal{L}_{\angle} = -\sum_{(\mathbf{x},y)\in B}\left(1 - \frac{\phi_{new}(\mathbf{x}) \cdot W_{old}^{(y)}}{\|\phi_{new}(\mathbf{x})\|\|W_{old}^{(y)}\|}\right)$$

[*] S. Ricci et al. ECCV 2024

ImageNet500/ImageNet1k dataset

Method	Case	mAP@1.0	CMC-1
Initial Model	ϕ_{old}/ϕ_{old}	32.33	43.53
Independent	$\phi^{I}_{new}/\phi_{old}$	00.14	00.12
	$\phi^{I}_{new}/\phi^{I}_{new}$	55.53	69.11
BCT [18]	$\phi^{BCT}_{new}/\phi_{old}$	35.81	48.56
	$\phi^{BCT}_{new}/\phi^{BCT}_{new}$	54.44	67.57
BT^2 [25]	$\phi^{BT^2}_{new}/\phi_{old}$	36.55	50.21
	$\phi^{BT^2}_{new}/\phi^{BT^2}_{new}$	55.65	67.75
OCA	$\phi^{OCA}_{new}/\phi_{old}$	**36.71**	**50.73**
	$\phi^{OCA}_{new}/\phi^{OCA}_{new}$	**56.82**	**69.73**

BT^2 loses compatibility when the embedding space is expanded with a large dimension.

Cifar50/100 dataset ablation on the extra dimention

Ext. Dim.	Method	Case	mAP@1.0	CMC-1
	Initial Model	ϕ_{old}/ϕ_{old}	23.32	31.32
	Independent	$\phi^{I}_{new}/\phi_{old}$	01.29	01.02
		$\phi^{I}_{new}/\phi^{I}_{new}$	45.35	56.75
+1	BT^2 [25]	$\phi^{BT^2}_{new}/\phi_{old}$	25.58	34.79
		$\phi^{BT^2}_{new}/\phi^{BT^2}_{new}$	44.42	59.84
	OCA	$\phi^{OCA}_{new}/\phi_{old}$	27.09	43.95
		$\phi^{OCA}_{new}/\phi^{OCA}_{new}$	50.87	61.04
+32	BT^2 [25]	$\phi^{BT^2}_{new}/\phi_{old}$	26.05	38.64
		$\phi^{BT^2}_{new}/\phi^{BT^2}_{new}$	50.36	61.77
	OCA	$\phi^{OCA}_{new}/\phi_{old}$	26.35	41.37
		$\phi^{OCA}_{new}/\phi^{OCA}_{new}$	52.06	62.02
+64	BT^2 [25]	$\phi^{BT^2}_{new}/\phi_{old}$	22.60	24.17
		$\phi^{BT^2}_{new}/\phi^{BT^2}_{new}$	50.36	62.87
	OCA	$\phi^{OCA}_{new}/\phi_{old}$	26.76	42.26
		$\phi^{OCA}_{new}/\phi^{OCA}_{new}$	51.69	61.03
+128	BT^2 [25]	$\phi^{BT^2}_{new}/\phi_{old}$	12.43	08.98
		$\phi^{BT^2}_{new}/\phi^{BT^2}_{new}$	48.64	62.37
	OCA	$\phi^{OCA}_{new}/\phi_{old}$	26.19	40.65
		$\phi^{OCA}_{new}/\phi^{OCA}_{new}$	52.12	61.82

Probability Simplex Projections [*]

DNNs can achieve compatibility with no training with additional losses, mapping or modifications of the network architecture.

Using the feature derived from softmax outputs for gallery and query.

Features from	R@1 w R18 (ImageNet1K)	R@1 w R152 (ImageNet1K V2)	R@1 w R18 galleries and R152 queries	Compatible?
Penultimate Layer	70.22	75.36	00.10	No
Logits	51.56	79.29	60.53	Yes
Softmax Probability Outputs	63.39	78.29	76.15	Yes

[*] S. Ricci et al., Under Review

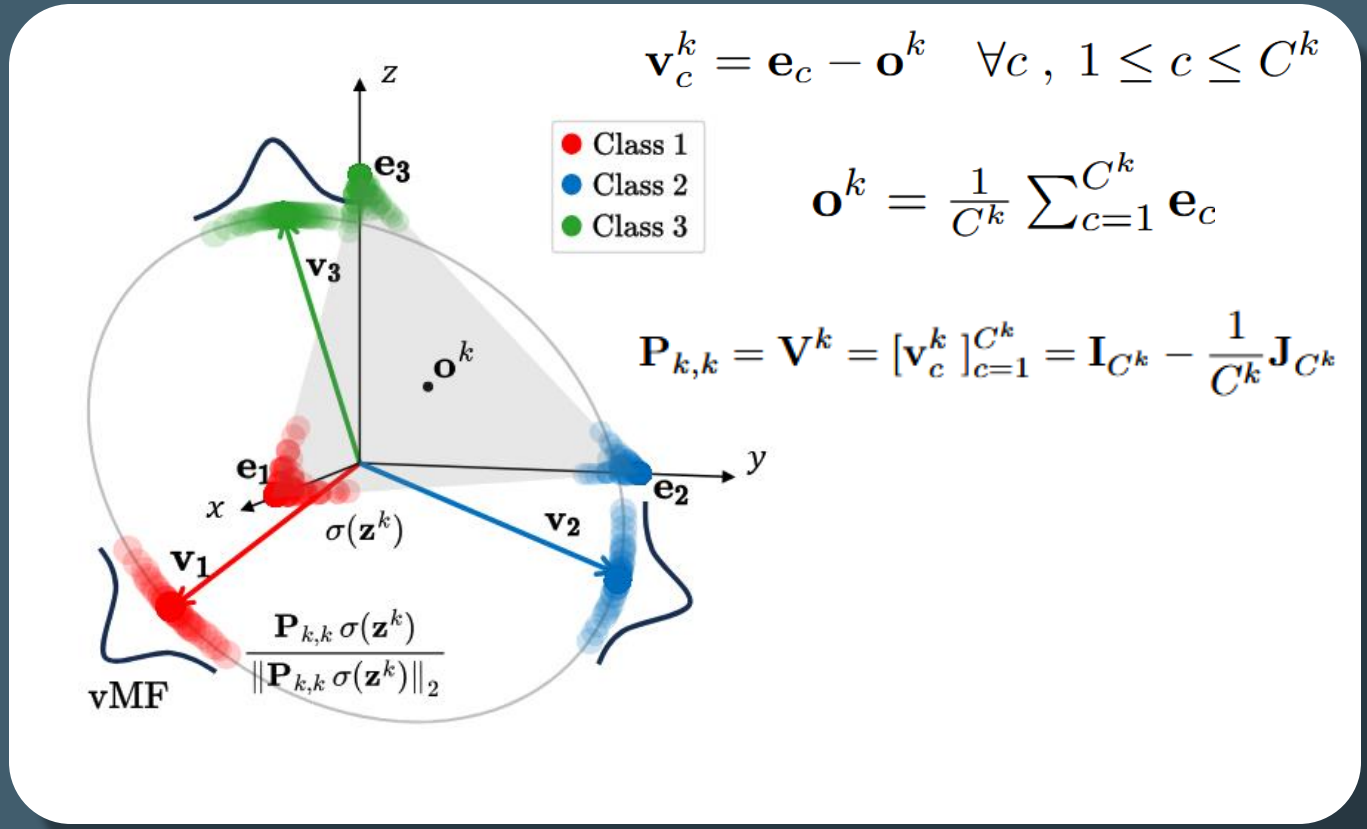
Probability Simplex Projections [*]

Probability Simplex (grey area) is a geometrical configuration such that its vertices e_i are the standard basis of the representation space and components of its points sum up to 1

Softmax probabilities converge to the **Probability Simplex** vertices e_i

Normalization of centered (through $P_{k,k}$) softmax outputs provides **hyperpherical simplex representation.**

Class prototypes of this representation are naturally aligned across updates and between independently trained models.

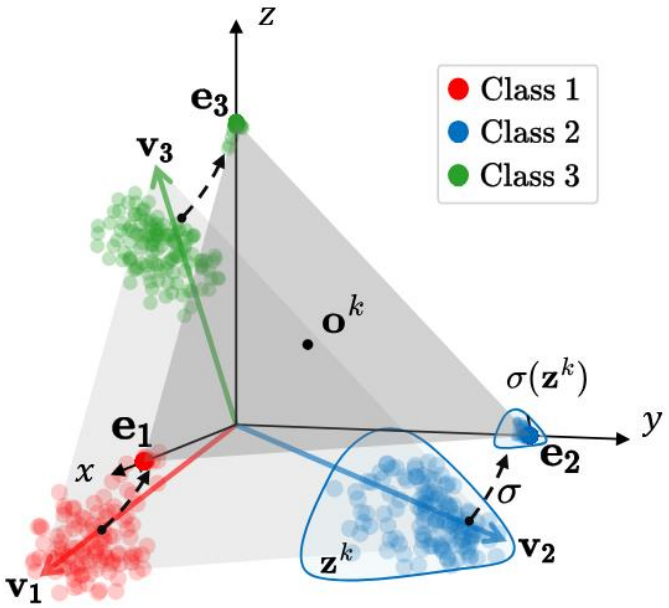


[*] S. Ricci et al., Under Review

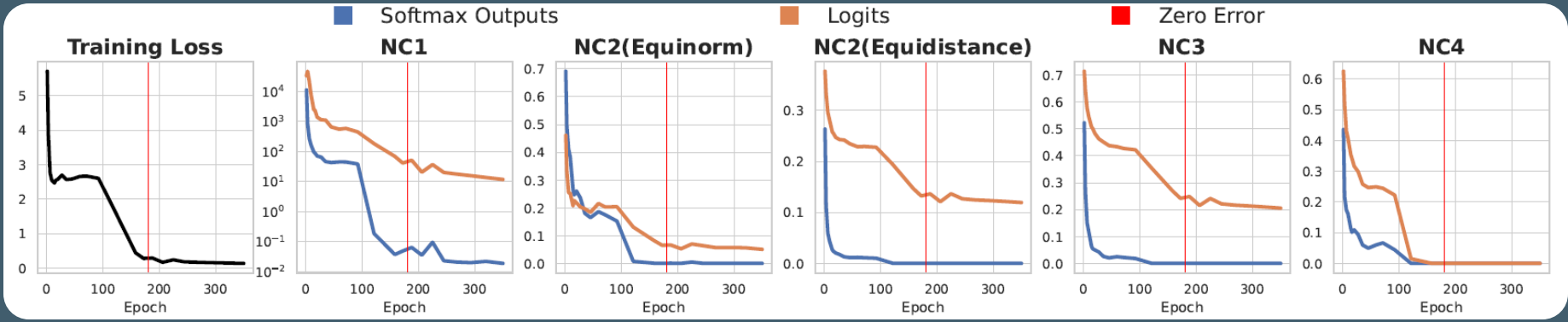
Logits Simplex Projections (LSP)

During training also logits configure in a simplex with the same prototype vectors …

Proposition 3.3. *During training, while softmax outputs converge to the vertices of the probability simplex, the corresponding logits vectors align to directions as defined in Eq. 2.*



… with more spread in the class features distribution
→ this leads to a more robust and generalizable representation

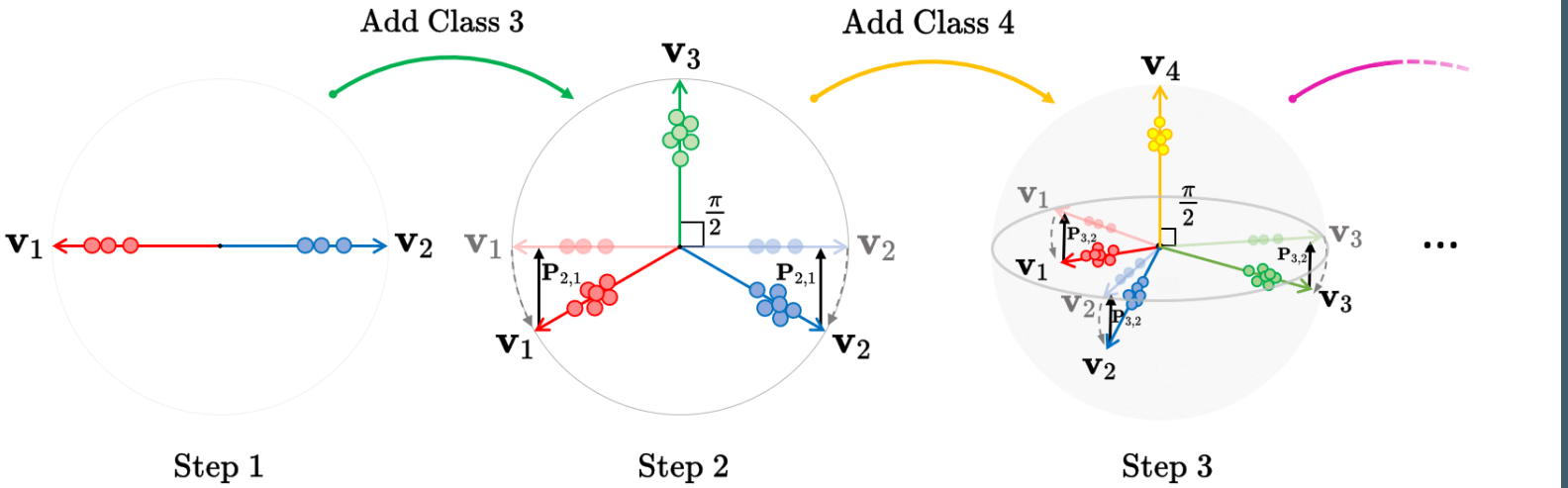


Deterministic Angular Correction Under Expanding Class Update

Hyperspherical simplex features configuration is **strictly related** to the number of classes (simplex configuration)

The class prototypes of the new classes are **orthogonal** with respect to the class prototypes from the previous step (**orthogonal feature space expansion**)

A **deterministic projection** matrix removes the angular misalignement.



$$\theta_c^{k \to t} = \arccos\left(\sqrt{\frac{C^k - 1}{C^k} \cdot \frac{C^t}{C^t - 1}}\right)$$

$$\mathbf{P}_{t,k} = [\mathbf{V}^k | \mathbf{0}]$$

$$\mathbf{V}_2 = \begin{bmatrix} \mathbf{e}_1 - \mathbf{o} \\ \mathbf{e}_2 - \mathbf{o} \end{bmatrix} = \begin{bmatrix} 1 - \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 1 - \frac{1}{2} \end{bmatrix}$$

$$\mathbf{P}_{3,2} = [\mathbf{V}_2 | \mathbf{0}] = \begin{bmatrix} 1 - \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 1 - \frac{1}{2} & 0 \end{bmatrix}$$

Probability Simplex Compatibility Theorem

Theorem 3.6 (***Compatibility Theorem***; Proof in Appendix F.). *Given model representations according to Eq. 4 and Eq. 6 obtained at different update steps, under the following assumptions:*

- *class features follow a von Mises-Fisher distribution* $\mathrm{vMF}(\mu, \kappa)$*;*
- *the concentration* κ *of class distributions increases across updates,*

the compatibility inequalities in Eq. 1a and Eq. 1b are verified in expectation.

PSP

$$\mathbf{h}^k = \frac{\mathbf{P}_{k,k}\,\sigma(\mathbf{z}^k)}{\|\mathbf{P}_{k,k}\,\sigma(\mathbf{z}^k)\|_2},$$

LSP

$$\mathbf{h}^k = \frac{\mathbf{z}^k}{\|\mathbf{z}^k\|_2},$$

$$\mathrm{dist}\big(\phi_{\mathrm{new}}(\mathbf{x}_i), \phi_{\mathrm{old}}(\mathbf{x}_j)\big) \leq \mathrm{dist}\big(\phi_{\mathrm{old}}(\mathbf{x}_i), \phi_{\mathrm{old}}(\mathbf{x}_j)\big) \quad \forall (i,j) \in \big\{(i,j) \,|\, y_i = y_j\big\}$$

and

$$\mathrm{dist}\big(\phi_{\mathrm{new}}(\mathbf{x}_i), \phi_{\mathrm{old}}(\mathbf{x}_j)\big) \geq \mathrm{dist}\big(\phi_{\mathrm{old}}(\mathbf{x}_i), \phi_{\mathrm{old}}(\mathbf{x}_j)\big) \quad \forall (i,j) \in \big\{(i,j) \,|\, y_i \neq y_j\big\},$$

EXPECTATION

$$\mathbb{E}\left[1 - (\mathbf{X}_1^k)^\top \mathbf{X}_2^t\right] \leq \mathbb{E}\left[1 - (\mathbf{X}_1^k)^\top \mathbf{X}_2^k\right]$$

for the case of same class

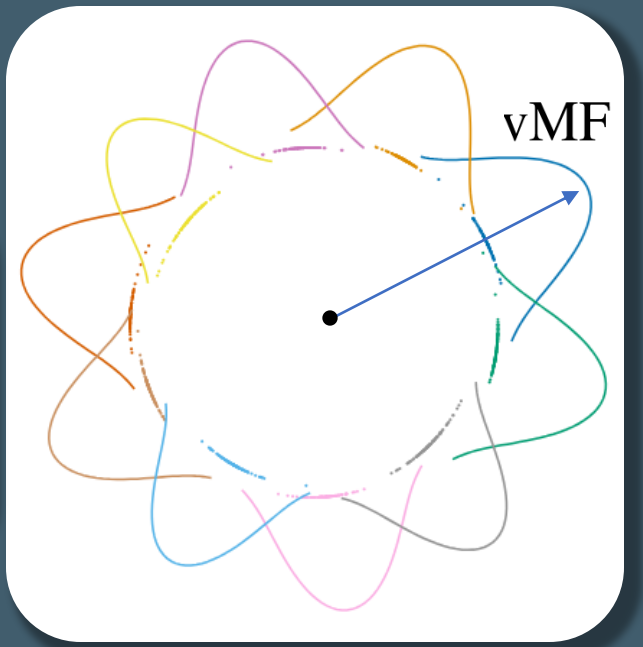
$$\mathbb{E}\left[1 - (\mathbf{X}_1^k)^\top \mathbf{X}_2^t\right] \geq \mathbb{E}\left[1 - (\mathbf{X}_1^k)^\top \mathbf{X}_2^k\right]$$

for the case of different classes

Theorem Proof

Features of each class **approximate the von Mises Fisher (vMF) distribution** on the hypersphere:

$$\mathbf{X} \sim \text{vMF}(\boldsymbol{\mu}_1, \kappa_1), \quad \mathbf{Y} \sim \text{vMF}(\boldsymbol{\mu}_2, \kappa_2) \quad \text{with } \mathbf{X}, \mathbf{Y} \in S^{d-1}$$



The verification of compatibility can be **analytically determined** according to a **closed-form solution:**

The expected cosine distance between features **X** and **Y** is a function of the **angle between the mean** of the vMF distributions (representing class prototypes) and their **concentrations** k_1, k_2

$$\mathbb{E}[1 - \cos\theta] = 1 - \mathbb{E}\left[\mathbf{X}^\top \mathbf{Y}\right] = 1 - m_d(\kappa_1)\, m_d(\kappa_2) \cos\alpha \quad \text{with } \cos\alpha = \boldsymbol{\mu}_1^\top \boldsymbol{\mu}_2$$

where $m_d(\kappa) = \dfrac{I_{\frac{d}{2}}(\kappa)}{I_{\frac{d}{2}-1}(\kappa)}$ and I_ν the modified Bessel function of the first kind of order ν

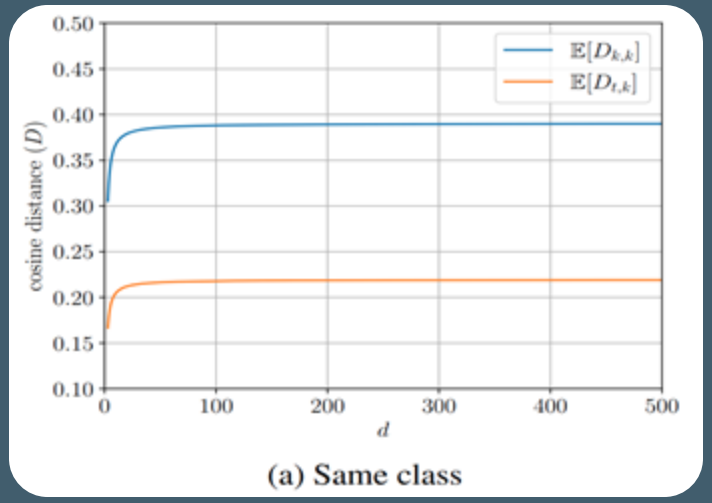
The vMF distribution of the updated model's class features have a **higher concentration**:

$k_2 > k_1 \Rightarrow m_d(k_2) > m_d(k_1)$

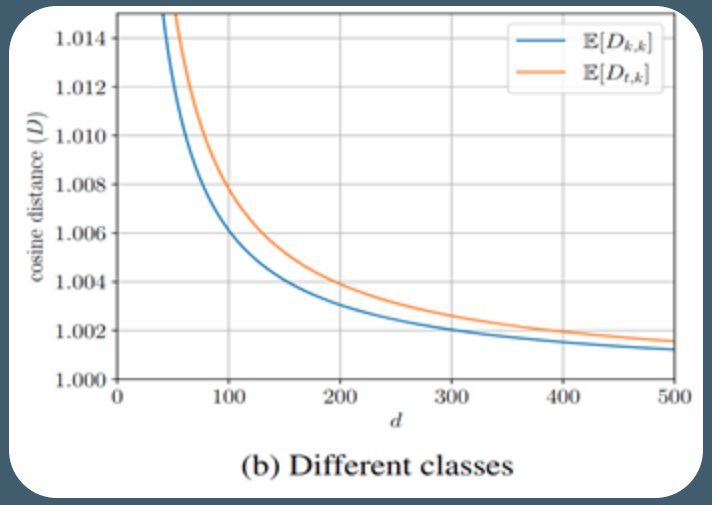
$$\mathbb{E}[1 - \cos\theta] = 1 - m_d(\kappa_1)\, m_d(\kappa_2) \cos\alpha$$

Features of the **same class** have aligned prototypes (i.e., cos α = 1) ⇒ the expected cosine distance between features **X** and **Y decreases**

For features from **different classes** the angle between prototypes is always **greater than π/2** (i.e., cos α < 0) ⇒ the expected cosine distance between features **X** and **Y increases**



(a) Same class



(b) Different classes

Compatibility Learning Metrics for Multiple Updates

From the Compatibility Matrix C:

Average Compatibility: the normalized count of the number of times Compatibility is assessed over T steps

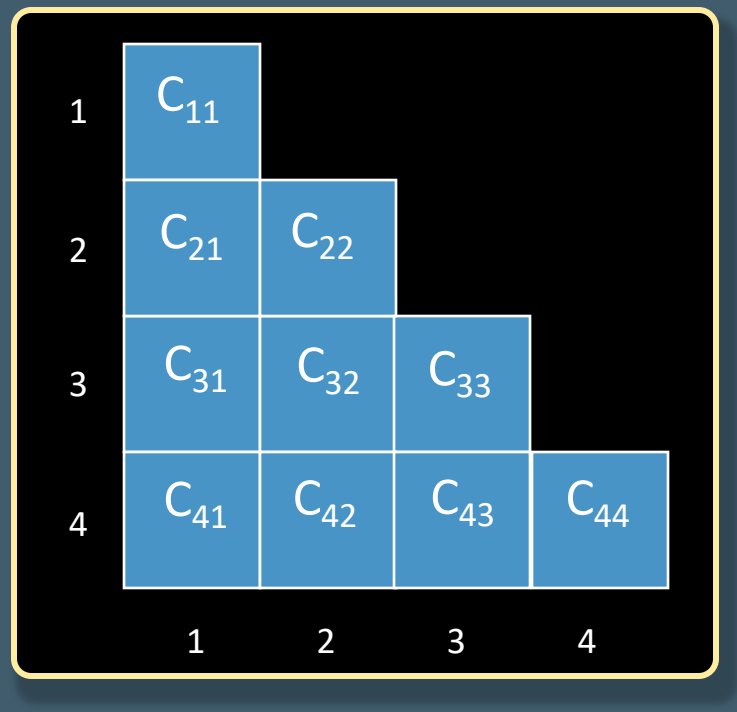
$$AC = \frac{2}{T(T-1)} \sum_{t=2}^{T} \sum_{k=1}^{t-1} \mathbb{1}\left(C_{t,k} > C_{k,k}\right),$$

Average Compatibility Accuracy: the average measure of Accuracy assessed over tasks that satisfy the Compatibility Criterion

$$ACA = \frac{2}{T(T-1)} \sum_{t=2}^{T} \sum_{k=1}^{t-1} C_{t,k} \quad \text{s.t. } C_{t,k} > C_{k,k}$$

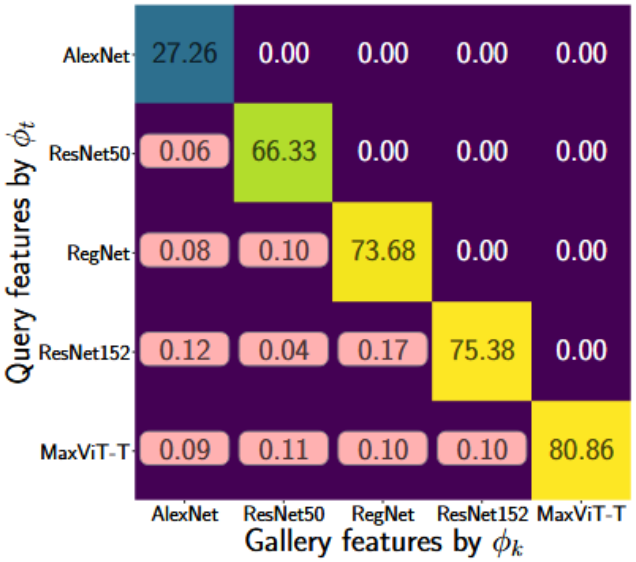
Average Accuracy: the average measure of Accuracy assessed over T steps

$$AA = \frac{2}{T(T+1)} \sum_{t=1}^{T} \sum_{k=1}^{t} C_{t,k},$$

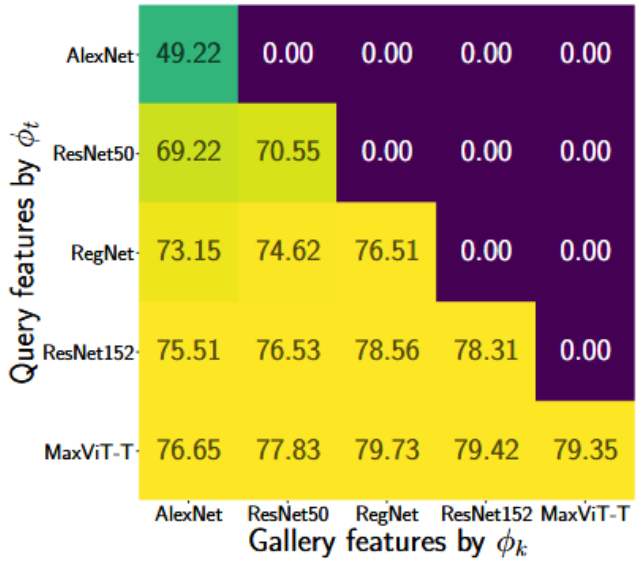


Compatibility Between Pretrained models from Pytorch Hub

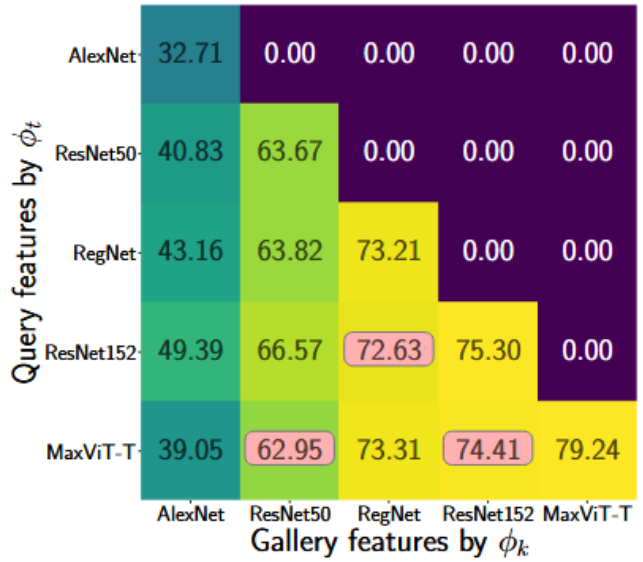
	AlexNet → R50 → RegNet → R152 → MaxViT_T						ViT-B-32 → ViT-B-16 → ViT-L-16					
	ImageNet1k			Places365			ImageNet1k			Places365		
METHOD	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
Baseline	0	21.63	0	0	8.38	0	0	38.63	0	0	15.10	0
PSP	**1**	**74.34**	**76.12**	**0.9**	15.92	**14.08**	**1**	**78.94**	**80.53**	**1**	18.31	**17.60**
LSP	0.7	60.68	37.61	0.2	**21.36**	4.06	0.67	71.82	45.91	0.33	**25.71**	7.99



Baseline



PSP



LSP

Cifar100/10 dataset

	2 steps			5 steps			20 steps			50 steps		
METHOD	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
Baseline	0	29.63	0	0	13.60	0	0	04.22	0	0	02.29	0
BCT [15]	**1**	40.64	35.51	0	**31.03**	0	0	22.66	0	0.01	15.78	0.84
RACT [41]	**1**	**44.19**	**45.09**	0.60	30.43	16.53	0.05	21.61	2.42	0	16.71	0
AdvBCT [35]	0	35.32	0	0.40	26.13	11.67	0.02	19.79	0.19	0	14.70	0.03
LCE [33]	**1**	**43.48**	**40.71**	0.10	**32.63**	04.63	0	20.95	0.25	0	13.81	0.03
CoReS [28]	**1**	40.94	35.21	0.30	30.65	12.29	0.12	**24.88**	4.61	0.11	**23.50**	3.91
PSP	**1**	36.31	29.05	**0.90**	26.04	**21.76**	**0.52**	20.56	**12.99**	**0.39**	19.42	**10.76**
LSP	**1**	41.14	36.38	**0.70**	30.36	**21.91**	**0.44**	**24.11**	**13.26**	**0.36**	**22.68**	**11.25**

Large scale datasets

	ImageNet1k						Google Landmark v2					
	2 steps			5 steps			2 steps			5 steps		
METHOD	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
Baseline	0	37.52	0	0	16.18	0	0	07.66	0	0	07.87	0
BCT [15]	**1**	48.81	**45.86**	0.20	34.96	5.28	**1**	09.37	08.22	0	08.46	0
RACT [41]	**1**	**55.17**	**51.16**	0.10	**40.94**	6.20	0	**11.01**	0	0	09.99	0
AdvBCT [35]	0	**53.07**	0	0	**43.91**	0	**1**	**11.12**	**09.67**	0	**10.73**	0
LCE [33]	0	47.22	0	0	32.67	0	**1**	09.41	08.03	0.40	**10.31**	3.80
CoReS [28]	0	46.11	0	0	37.67	0	oom	oom	oom	oom	oom	oom
PSP	**1**	49.10	39.80	**1**	35.20	**31.09**	**1**	08.74	08.24	**0.90**	08.53	**7.21**
↪ w top-256	**1**	49.11	39.81	**1**	35.20	**31.09**	**1**	08.74	08.24	**0.90**	08.53	**7.21**
LSP	**1**	50.72	45.62	0.50	38.03	14.28	**1**	09.31	**08.36**	**0.80**	09.36	**7.34**
↪ w top-256	**1**	50.34	45.84	**0.70**	37.96	**24.85**	**1**	09.31	**08.36**	**0.80**	09.36	**7.34**

Looking at the code

PSP-LSP Compatibility

Google Colab link

Jupyter Notebook link

Coffee Break ☕

30’

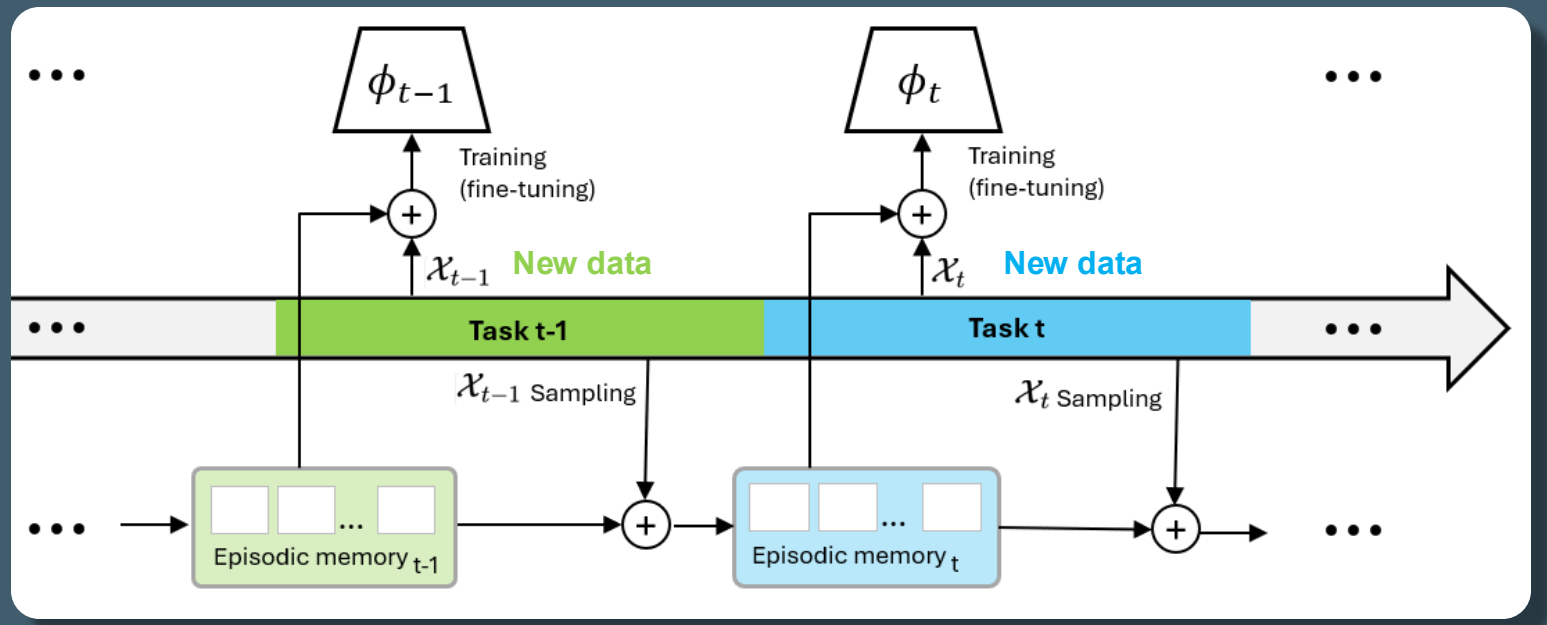
Compatibility

Issues and Solutions

Compatibility in Continual Learning

Compatibility training using old and new data **is time-consuming** as **t**raining datasets are larger and larger and models more and more complex.
It **might be unfeasible** if previous training data are no more available

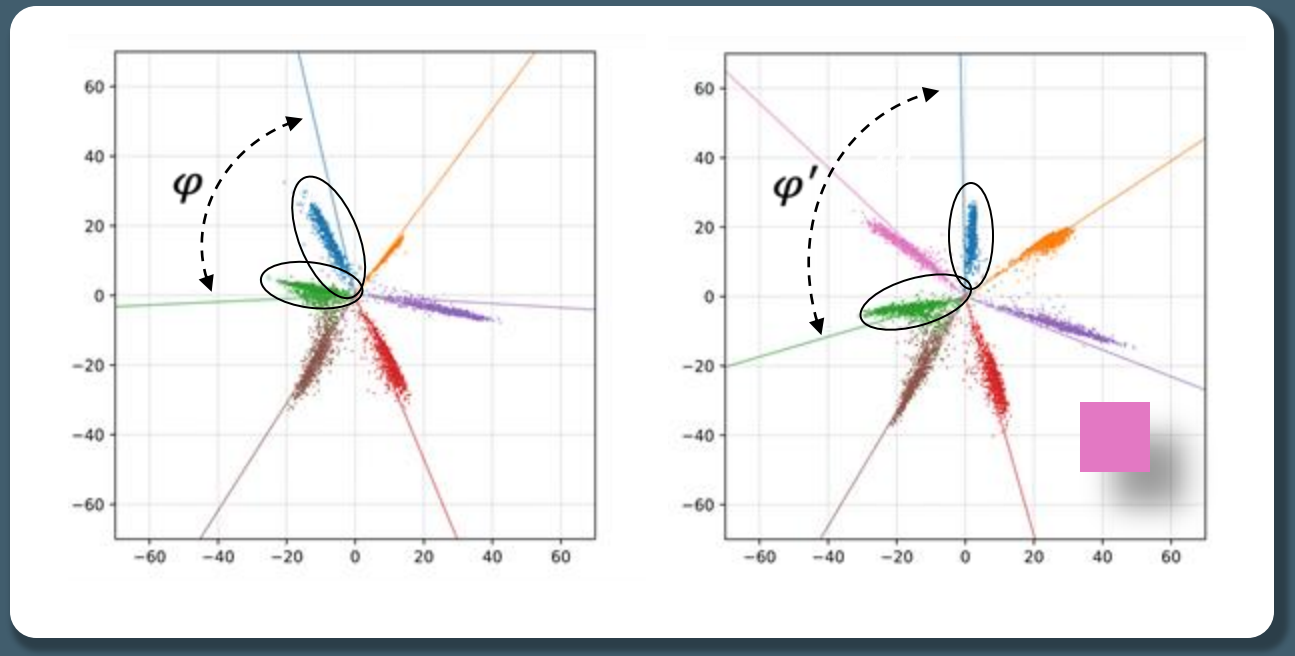
Compatibility in Continual Learning: learning compatible representation where at each step the model is trained using **only the new data** and a **small replay memory**



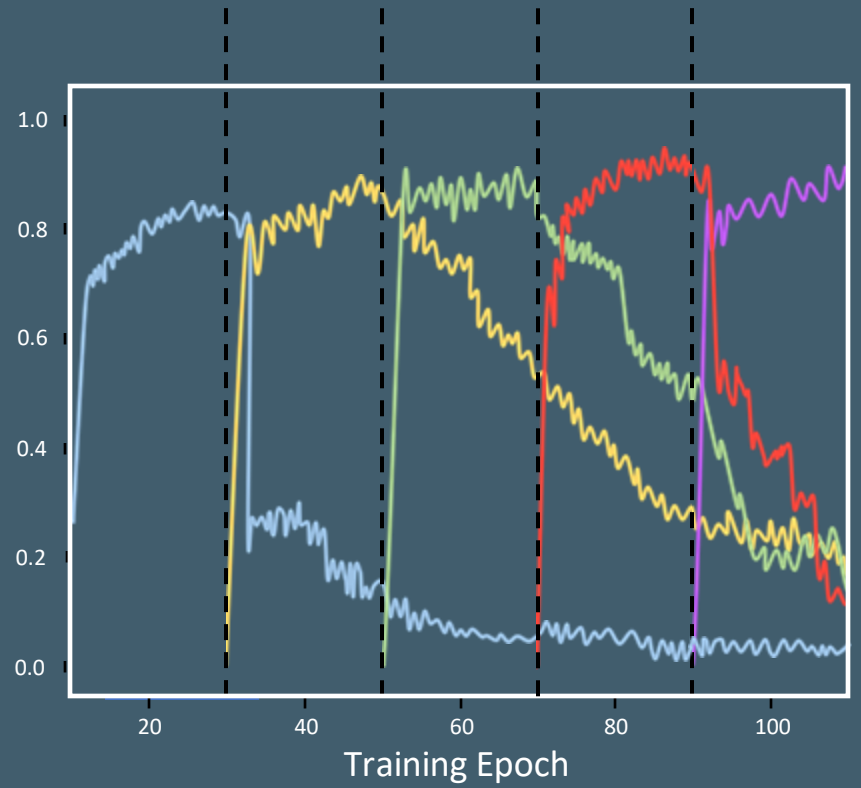
picture
authored by N. Biondi

Main Issues

Updating a DNN model with novel classes changes the internal feature representation



Performance degradation occurs when new knowledge is assimilated (*catastrophic forgetting*)



pictures
authored by F. Pernici and N. Biondi

Proposals for Solutions

Regularization-based approaches:
Use distillation and experience replay to mitigate forgetting and additional losses to preserve Compatibility at model upgrading

- ✓ CVS (CVPR2023)
- ✓ C^2R (CVPR2024)

Mapping-based approaches:
Use distillation and experience replay to mitigate forgetting and finds compatible mappings between the previous and upgraded model

- ✓ FAN (ECCV2020)
- ✓ HBCT (ICML2025)

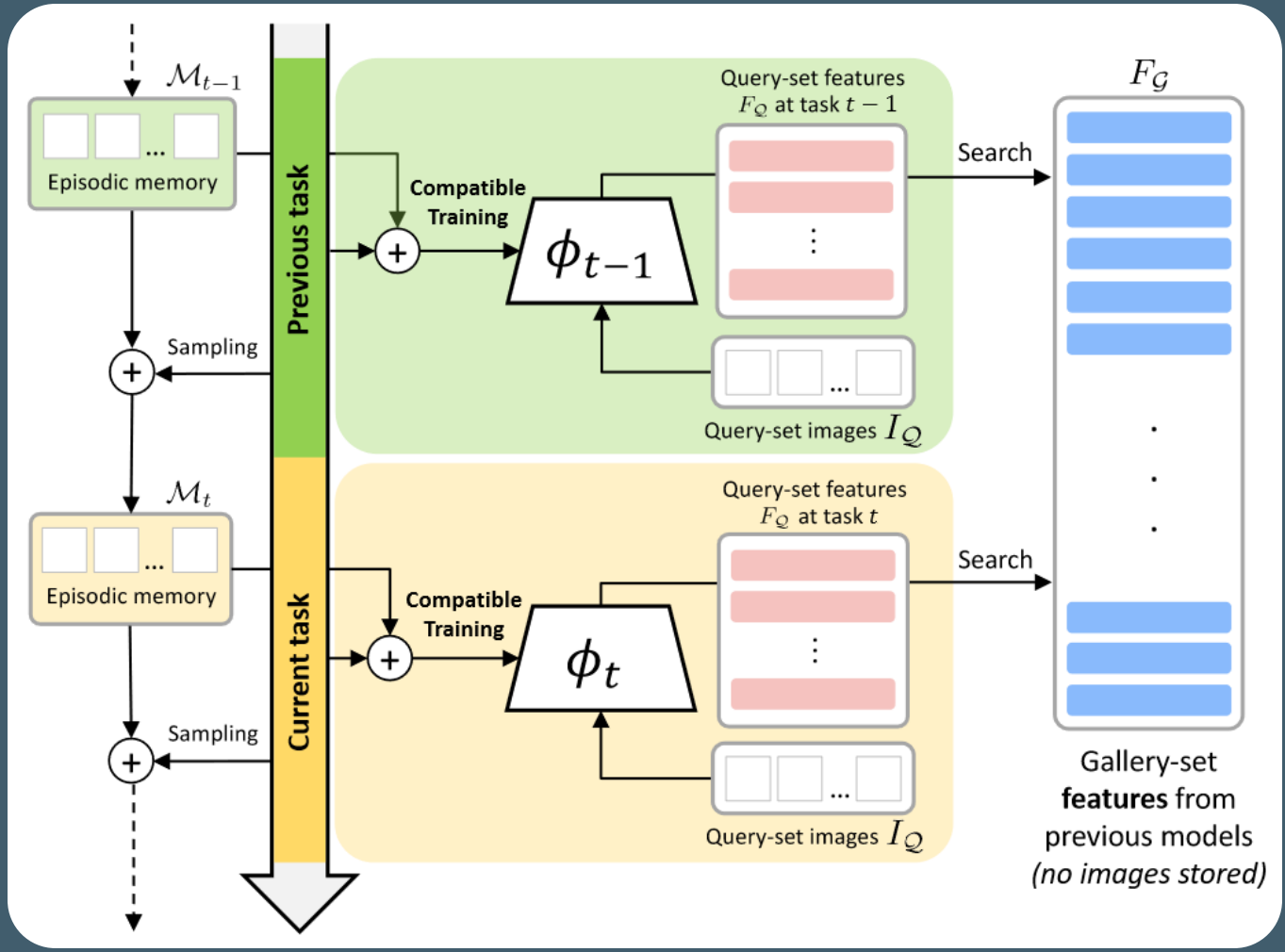
Approaches with **architectural changes**:
Use distillation and experience replay to mitigate forgetting and impose constrains on the network architecture that force feature alignment

- ✓ d-Simplex-FD (ACM TOMM2023)
- ✓ d-Simplex-HOC (CVPR2024)

d-Simplex with Feature Distillation

Stationarity (**global**): d-Simplex classifier [3,16]. Features do not change their geometric configuration during model updates

Stationarity (**local**): feature distillation [17]. Used to both reduce catastrophic forgetting and impose additional stationarity from the previously learned model



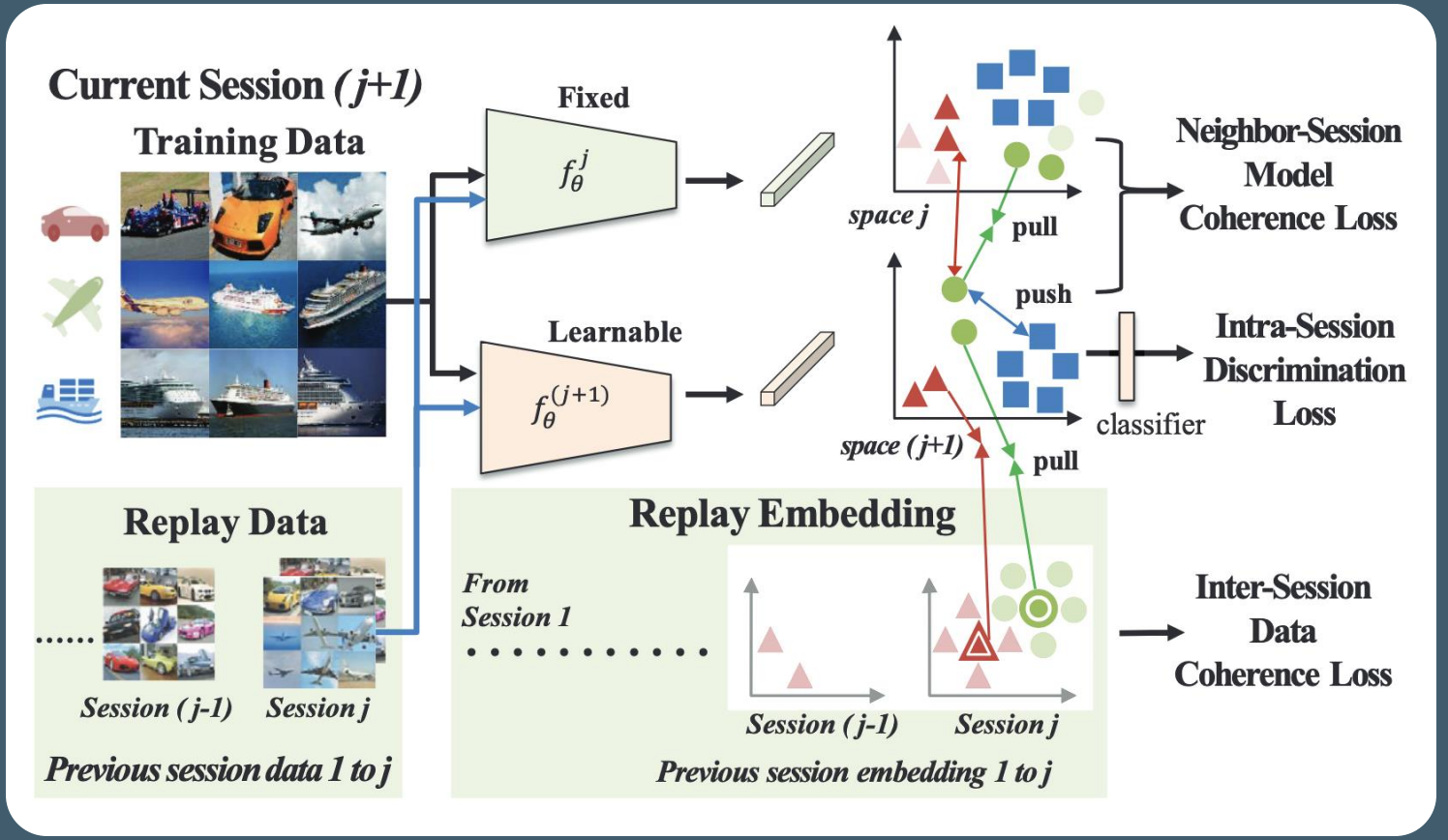
$$\mathcal{L}_{\text{FD}}^{\mathcal{M}} = \frac{1}{|\mathcal{M}_t|} \sum_{\mathbf{x}_i \in \mathcal{M}_t} \left(1 - \frac{\phi_t(\mathbf{x}_i) \cdot \phi_{t-1}(\mathbf{x}_i)}{\|\phi_t(\mathbf{x}_i)\| \, \|\phi_{t-1}(\mathbf{x}_i)\|} \right)$$

[3] Pernici *et al.* ICPR2020
[16] Biondi *et al.* TPAMI2023
[17] Romero *et al.* arXiv 2014

Continual Learning for Visual Search with Backward Consistent Feature Embedding

The three losses for a continual learner:

- ✓ **intra-session discrimination** loss to learn discriminative representation within the current-session data
- ✓ **neighbor session model coherence** loss to regulate the current model with the previous session model for backward compatible embedding
- ✓ **inter-session data coherence** with the current-session data and replayed embedding of all previous sessions for long term embedding consistency

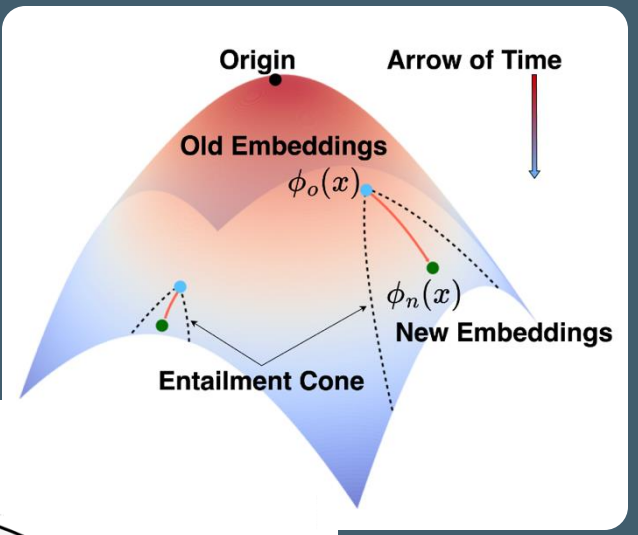


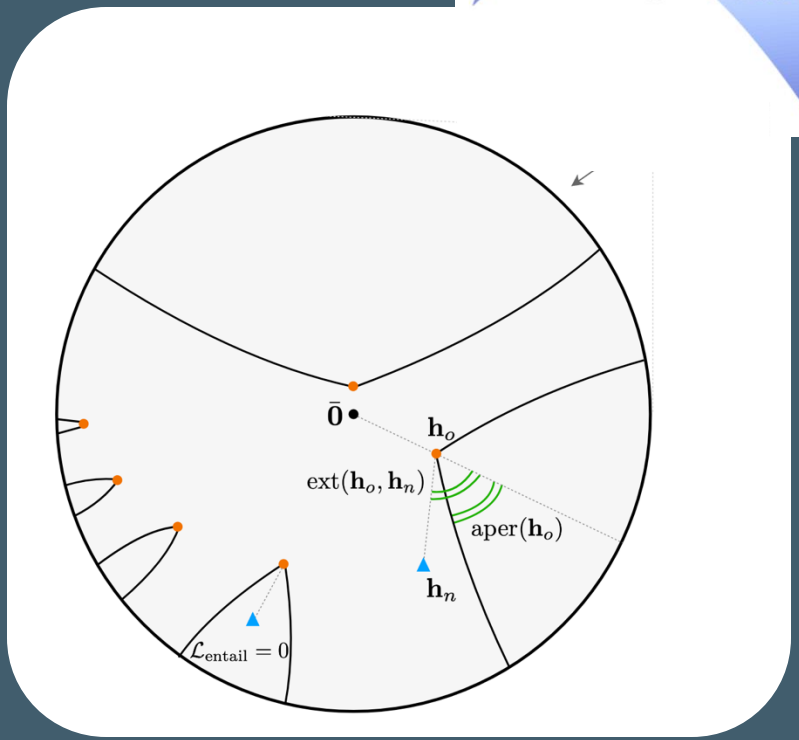
Learning Along the Arrow of Time: Hyperbolic Geometry for Backward-Compatible Representation Learning

The entailment cone enforces a partial-order constraint between the old and new models

- ✓ This constraint requires the new embeddings to remain within the **entailment cone** of their older counterparts, maintaining **backward compatibility** while embracing the improved expressiveness brought by new data or architectures

A **hyperbolic contrastive loss** is further introduced that dynamically adjusts alignment weights based on the uncertainty levels of old embeddings, enabling more adaptive and reliable alignment across model generations

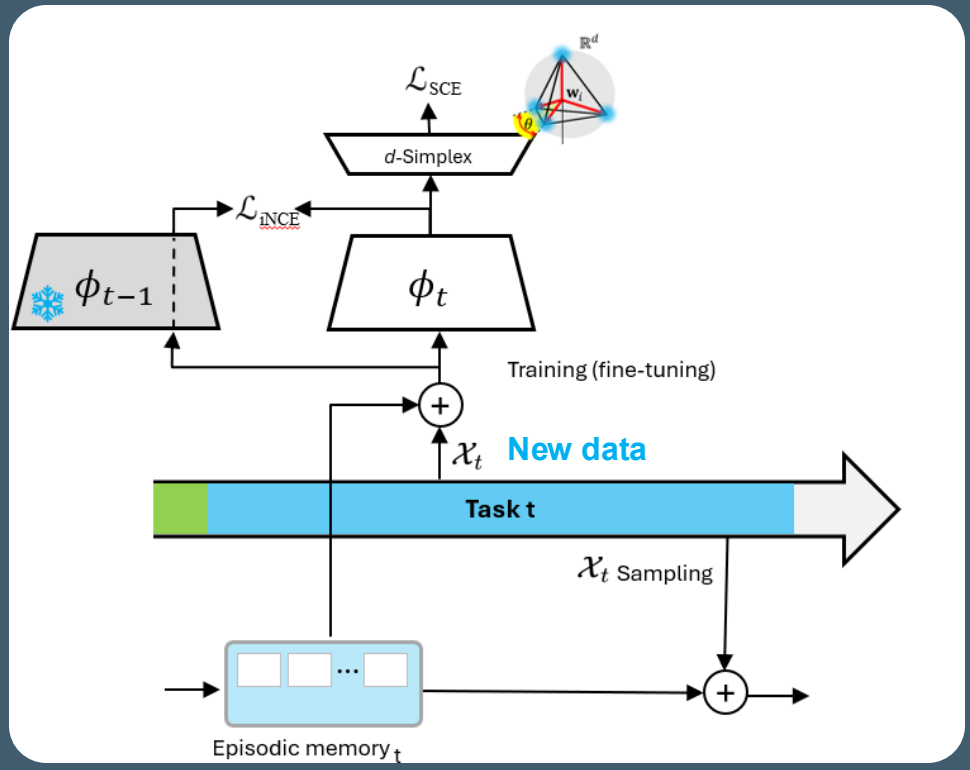




d-Simplex with Higher Order Compatibility Loss (d-Simplex HOC)[*]

Training according to d-Simplex using Cross-Entropy loss aligns features of each class to the first-order moment of their distribution. Cannot capture **higher-order dependencies** between model updates

- ✓ **d-Simplex classifier** to keep feature geometric configuration stable during model updates
- ✓ **Higher-Order Compatibility (HOC) loss** combines Cross-Entropy loss and Contrastive loss



[*] N. Biondi et al., CVPR 2024

HOC loss

$$\mathcal{L}_{\text{HOC}}(\phi_t) = \lambda\,\mathcal{L}_{\text{SCE}}(\phi_t) + (1-\lambda)\,\mathcal{L}_{i\text{NCE}}(\phi_t, \phi_{t-1}), \quad \lambda \in [0,1]$$

$$\mathcal{L}_{i\text{NCE}}(\phi_t, \phi_{t-1}) = -\sum_{B} \log\left(\frac{\Delta\big(\phi_{t-1}(\mathbf{x}_i), \phi_t(\mathbf{x}_i)\big)}{\sum_{j\neq i}\Delta\big(\phi_{t-1}(\mathbf{x}_i), \phi_t(\mathbf{x}_j)\big)}\right)$$

where $\Delta\big(\phi_{t-1}(\mathbf{x}_i), \phi_t(\mathbf{x}_j)\big) = \exp\left(\rho\ \frac{\phi_{t-1}(\mathbf{x}_i)\phi_t(\mathbf{x}_j)}{||\phi_{t-1}(\mathbf{x}_i)||\,||\phi_t(\mathbf{x}_j))||}\right)$

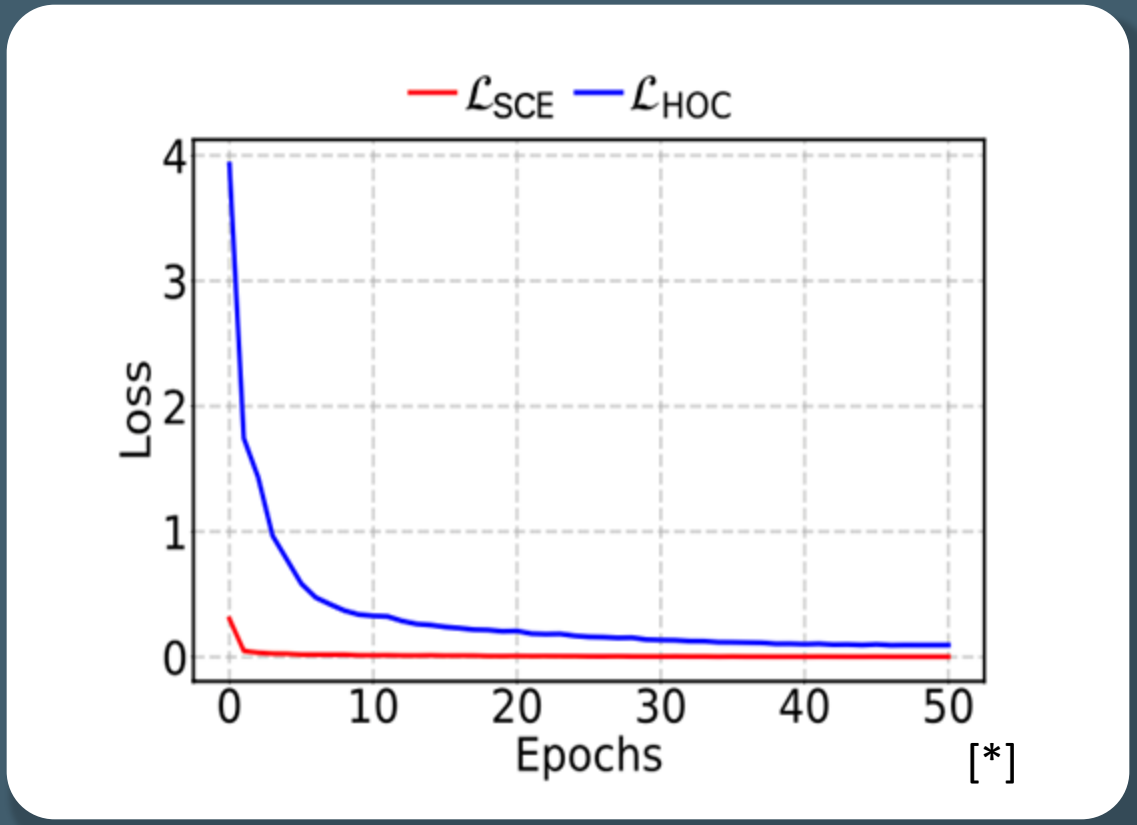
A **convex combination** of the Cross-Entropy loss and a Contrastive loss

Contrastive loss based on the ρ-scaled cosine similarity between $\varphi_{t-1}(\mathbf{x}_i)$ and $\varphi_t(\mathbf{x}_j)$

The Contrastive loss approximates the **mutual information** between ϕ_t and ϕ_{t-1}
It captures **higher-order dependencies** in the representation between model updates

HOC Loss vs Cross Entropy Loss

MNIST dataset



LeNet++ trained with the d-Simplex fixed classifier on
5 MNIST classes and fine-tuned with the other 5 classes

Rapid convergence when using Cross-Entropy loss may lead to the **model's inability to learn higher-order dependencies** between consecutive model updates

[*] N. Biondi et al., IEEE TPAMI 2024 (sub)

HOC Loss Equivalence with Compatibility

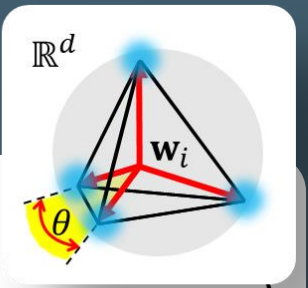
While capturing higher-order dependencies, the HOC loss **preserves compatibility** between learned representations

Proposition 1 (Loss Equivalence). *Let ϕ_t, ϕ_{t-1} be two representation models learned according to a d-Simplex fixed classifier where the updated model ϕ_t is obtained by fine-tuning ϕ_{t-1}. Then, training ϕ_t with the HOC loss is equivalent to training ϕ_t using cross-entropy loss under compatibility constraints.* [*]

[*] N. Biondi et al., CVPR 2024

Stationarity and Higher-Order Alignment

- ✓ From [17], the constrained problem of using Simplex CE with compatibility inequalities as constraints is equivalent to a:
  - ✓ convex combination of the CE loss and Kullback-Leibler loss
- ✓ … this means that it captures higher-order dependencies in the model training process



$$\mathcal{L}_{\text{SCE}}(\phi_t) = \\ = -\sum_{B} \log \left(\frac{\exp\left(\mathbf{W}_{y_i}^\top \phi_t(\mathbf{x}_i)\right)}{\sum_{j=1}^{K_t} \exp\left(\mathbf{W}_j^\top \phi_t(\mathbf{x}_i)\right) + \sum_{j=K_t+1}^{K} \exp\left(\mathbf{W}_j^\top \phi_t(\mathbf{x}_i)\right)} \right)$$

$$\begin{aligned} \underset{\phi_t}{\text{argmin}} \quad & \mathcal{L}_{\text{SCE}}(\phi_t) \\ \text{s.t.} \quad & \mathrm{d}\big(\phi_k(\mathbf{x}_i), \phi_t(\mathbf{x}_j)\big) - \mathrm{d}\big(\phi_k(\mathbf{x}_i), \phi_k(\mathbf{x}_j)\big) \leq 0 \\ & \qquad \forall\, (\mathbf{x}_i, \mathbf{x}_j) \text{ such that } y_i = y_j \\ & \mathrm{d}\big(\phi_k(\mathbf{x}_i), \phi_t(\mathbf{x}_j)\big) - \mathrm{d}\big(\phi_k(\mathbf{x}_i), \phi_k(\mathbf{x}_j)\big) \geq 0 \\ & \qquad \forall\, (\mathbf{x}_i, \mathbf{x}_j) \text{ such that } y_i \neq y_j \end{aligned}$$

$$\mathcal{L}_{\text{HOC}}(\phi_t) = \lambda \mathcal{L}_{\text{SCE}}(\phi_t) + (1-\lambda)\, \mathcal{L}_{\text{NCE}}(\phi_t, \phi_{t-1}), \\ \text{with} \quad \lambda \in [0, 1]$$

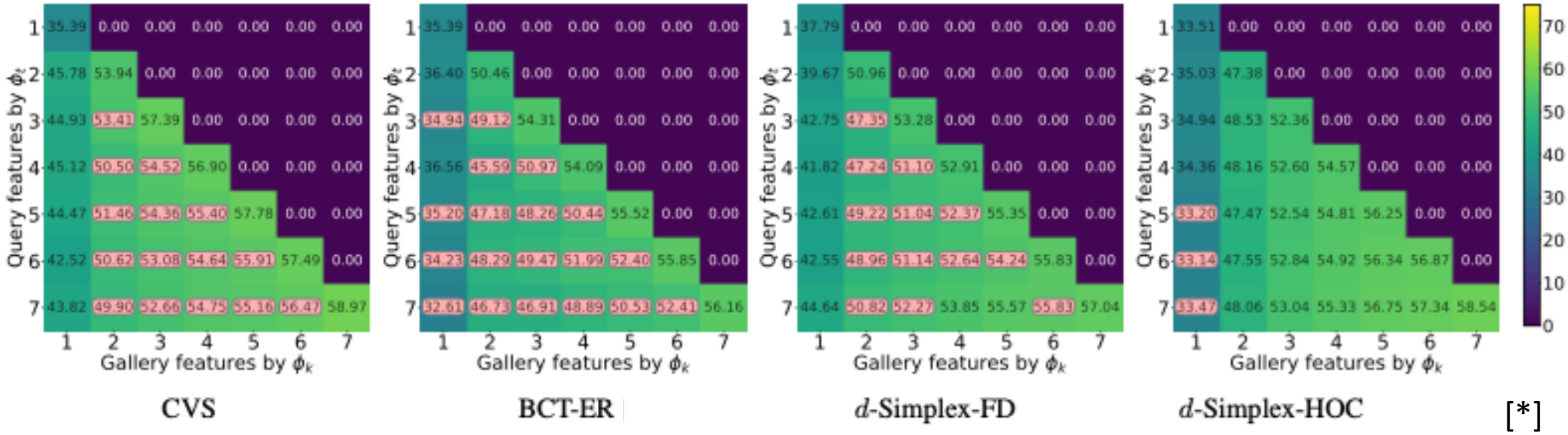
[17] Jiang *et al.* ICLR2021

Comparative Evaluation

ResNet32
CIFAR100/10 Dataset

$$\underbrace{M\left(\phi^{\mathcal{Q}}_{\text{new}},\ \phi^{\mathcal{G}}_{\text{old}}\right)}_{\text{Cross-Test}} > \underbrace{M\left(\phi^{\mathcal{Q}}_{\text{old}},\ \phi^{\mathcal{G}}_{\text{old}}\right)}_{\text{Self-Test}}$$

open set



[*]

[*] N. Biondi et al., IEEE TPAMI 2024 (sub)

Compatible Continual Learning

ResNet32 **CIFAR100/10 dataset**

100 training classes: init 10 classes [*]

Method	2 tasks			7 tasks			16 tasks			31 tasks		
	AC	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
ER baseline	1.00	46.67	39.25	0.19	46.75	7.79	0.04	44.91	1.69	0.02	40.78	0.93
FAN [19]	1.00	48.37	43.87	0.14	47.59	5.63	0.15	46.10	6.69	0.07	43.12	3.42
BCT-ER	1.00	47.11	41.14	0.10	46.82	3.48	×	43.63	×	0.01	41.96	0.40
LCE-ER	1.00	49.78	50.02	0.24	45.15	10.38	0.28	41.98	12.61	0.14	38.45	5.93
AdvBCT-ER	1.00	46.28	39.71	×	35.66	×	0.01	34.89	0.39	0.02	33.93	1.04
CVS [7]	1.00	**51.31**	**52.89**	0.29	**51.69**	12.70	0.13	**49.38**	5.65	0.07	46.33	2.93
d-Simplex-FD [20]	1.00	48.76	44.96	0.38	49.67	17.31	0.45	48.42	21.32	0.34	44.82	15.37
d-Simplex-HOC (this paper)	1.00	49.98	48.48	**0.86**	48.21	**42.41**	**0.90**	47.56	**43.20**	**0.49**	**47.34**	**23.44**

M: Search accuracy 1:N search

ResNet18 **TinyImageNet /ImageNet20 dataset**

200 training classes; init 50 classes

Method	2 tasks			7 tasks			16 tasks			31 tasks		
	AC	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
ER baseline	1.00	34.87	34.87	×	24.63	×	×	21.87	×	<0.01	19.15	0.05
FAN [19]	1.00	33.94	33.94	×	33.94	×	×	22.94	×	<0.01	20.11	0.10
BCT-ER	1.00	34.67	34.67	×	24.27	×	0.01	22.77	0.22	<0.01	19.67	0.04
LCE-ER	1.00	34.21	34.21	×	22.19	×	×	16.82	×	0.02	14.80	0.34
AdvBCT-ER	1.00	34.32	34.32	×	19.77	×	×	19.02	×	<0.01	16.61	0.04
CVS [7]	1.00	**37.14**	**37.14**	0.10	**33.20**	3.29	0.05	29.79	1.62	0.01	24.71	0.35
d-Simplex-FD [20]	×	32.20	×	0.14	30.52	4.43	0.05	29.17	1.55	0.02	27.09	0.44
d-Simplex-HOC (this paper)	1.00	36.85	36.85	**0.81**	31.45	**25.96**	**0.82**	**31.91**	**26.72**	**0.77**	**32.35**	**25.14**

AC Average Compatibility: Normalized count of Compatibility over T tasks

AA Average Accuracy: Average Accuracy over T tasks

ACA Average Compatibility Accuracy: Average Accuracy over tasks that satisfy the Compatibility Criterion

ResNet50 pre-trained Imagenet 1k **CUB 180/20 dataset**

180 training classes; init 60 classes)

Method	2 tasks			7 tasks			16 tasks			31 tasks		
	AC	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*	*AC*	*AA*	*ACA*
ER baseline	1.00	**46.83**	46.26	0.48	43.77	20.80	0.25	42.85	10.55	0.07	38.54	3.08
FAN [19]	1.00	45.58	43.71	0.62	43.77	26.25	0.35	41.46	14.49	0.10	36.65	4.16
BCT-ER	1.00	44.90	43.71	0.62	**45.15**	27.06	0.26	41.37	11.03	0.08	38.26	4.00
LCE-ER	1.00	46.54	**47.62**	0.10	39.61	3.91	0.09	35.25	3.47	0.05	30.95	1.82
AdvBCT-ER	1.00	43.37	41.67	0.38	40.55	15.45	0.13	37.95	4.76	0.07	36.25	2.73
CVS [7]	1.00	45.58	46.09	0.62	44.75	27.28	0.23	**43.12**	9.87	0.14	39.67	5.96
d-Simplex-FD [20]	1.00	39.40	36.74	0.86	39.74	35.08	0.60	39.09	24.03	0.55	39.42	21.97
d-Simplex-HOC (this paper)	1.00	41.61	42.52	**0.90**	41.50	**37.72**	**0.73**	41.17	**30.01**	**0.60**	**42.01**	**25.46**

Biondi et al., IEEE TPAMI 2024 (sub)

A Paradigm Shift in AI

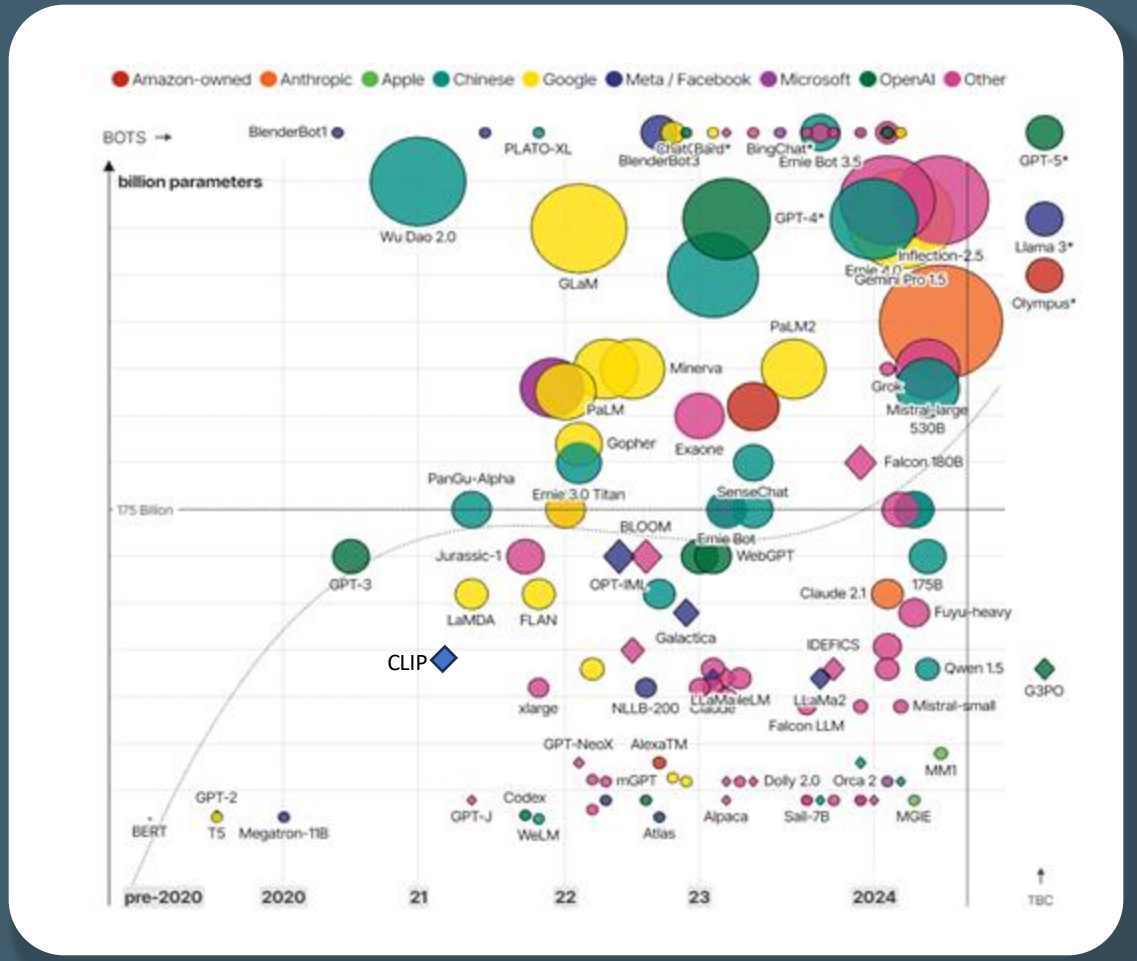
Foundation Models

Pre-trained Large Models

Large models pre-trained by third parties are more and more a fundamental building block for applications

The representation from the **Large Model's encoder** is exploited to develop specific applications

Used for both **Discriminative** and **Generative** tasks



picture authored by D. McCandless

Foundation Models updating

When updated Foundation Models may exhibit distinct issues that involve Compatibility

User–side

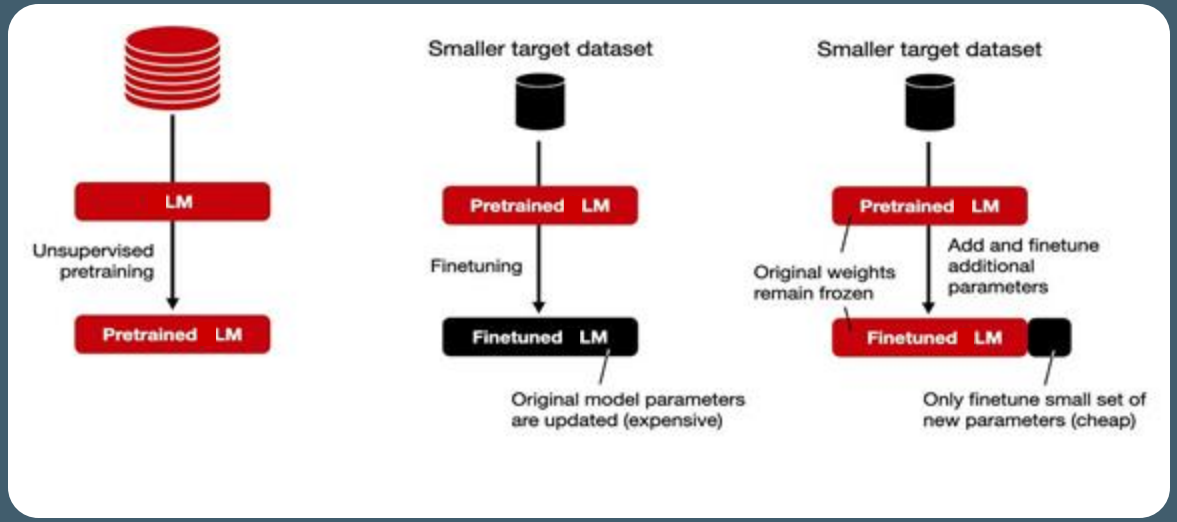
The now-standard practice is downloading and fine-tuning representations from the large pre-trained models. Compatibility with the user's fine-tuned version should ***allow seamless replacement***

Third parties-side

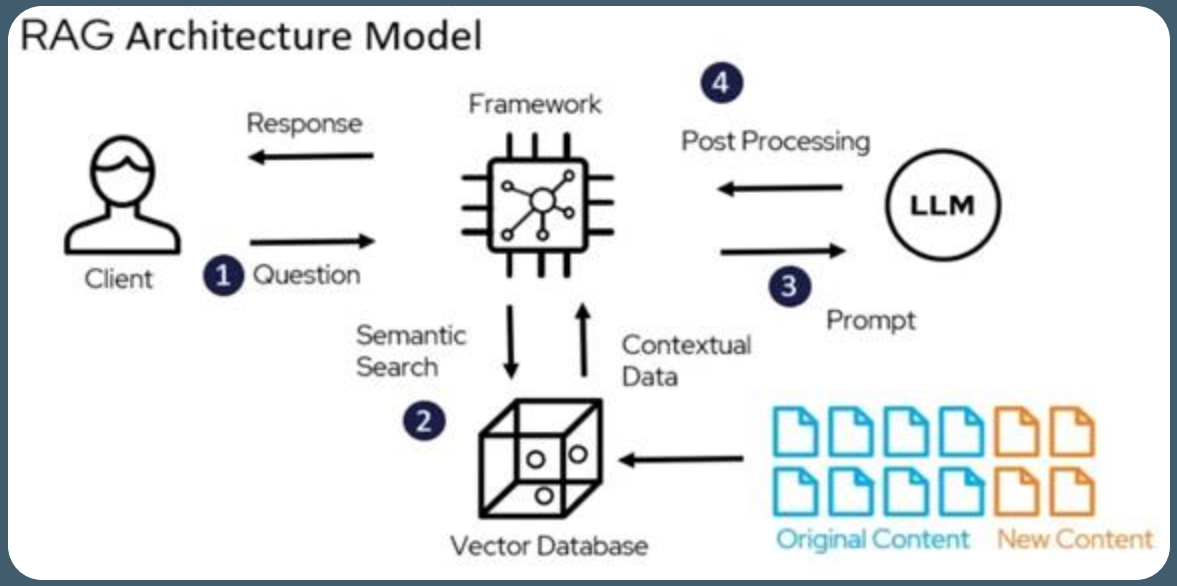
When the foundation model is updated Compatibility with the previous versions of the model should ***maintain consistency*** *and minimize disruption of the user experience*

User-side Adaptation

With a lightweight **adapter** or a **fine-tuned version** of the foundation model
(mostly for Discriminative tasks)



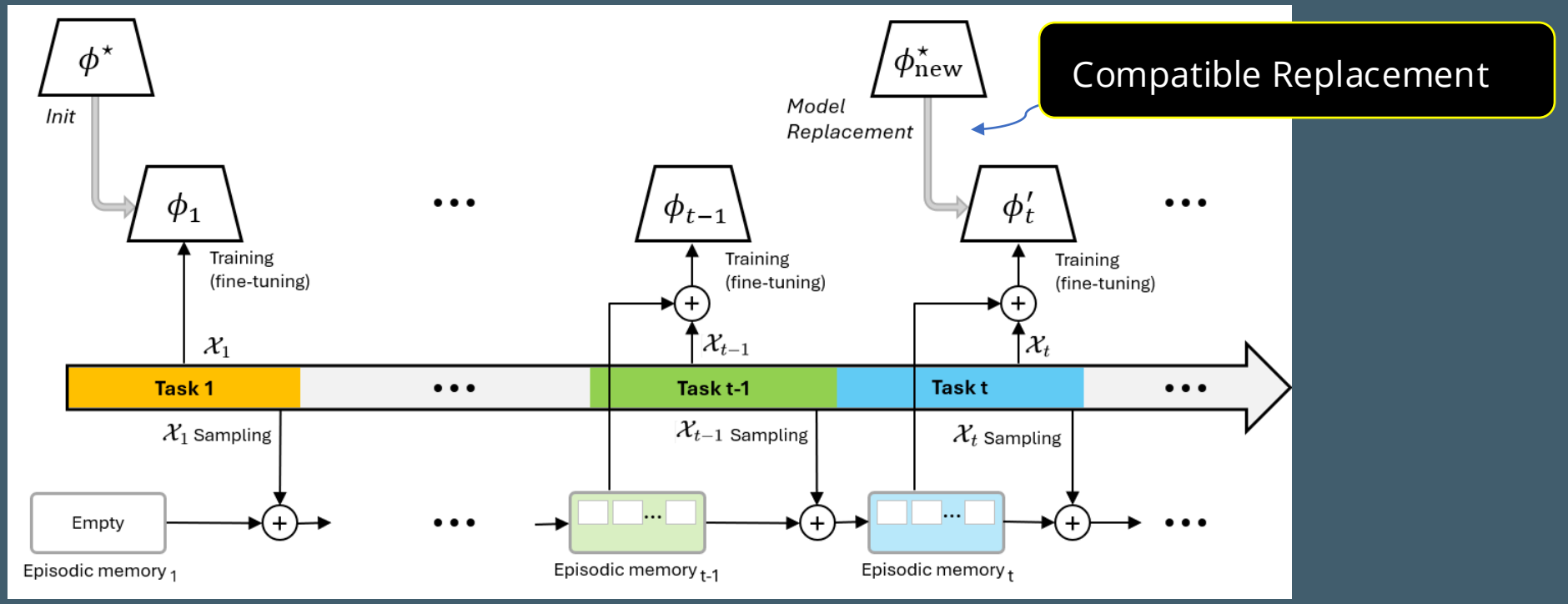
With **RAG (Retrieval Augmented Generation) -based applications** where the context of the foundation model is expanded with a set of user's private documents
(mostly for Generative tasks)



User-side Compatibility

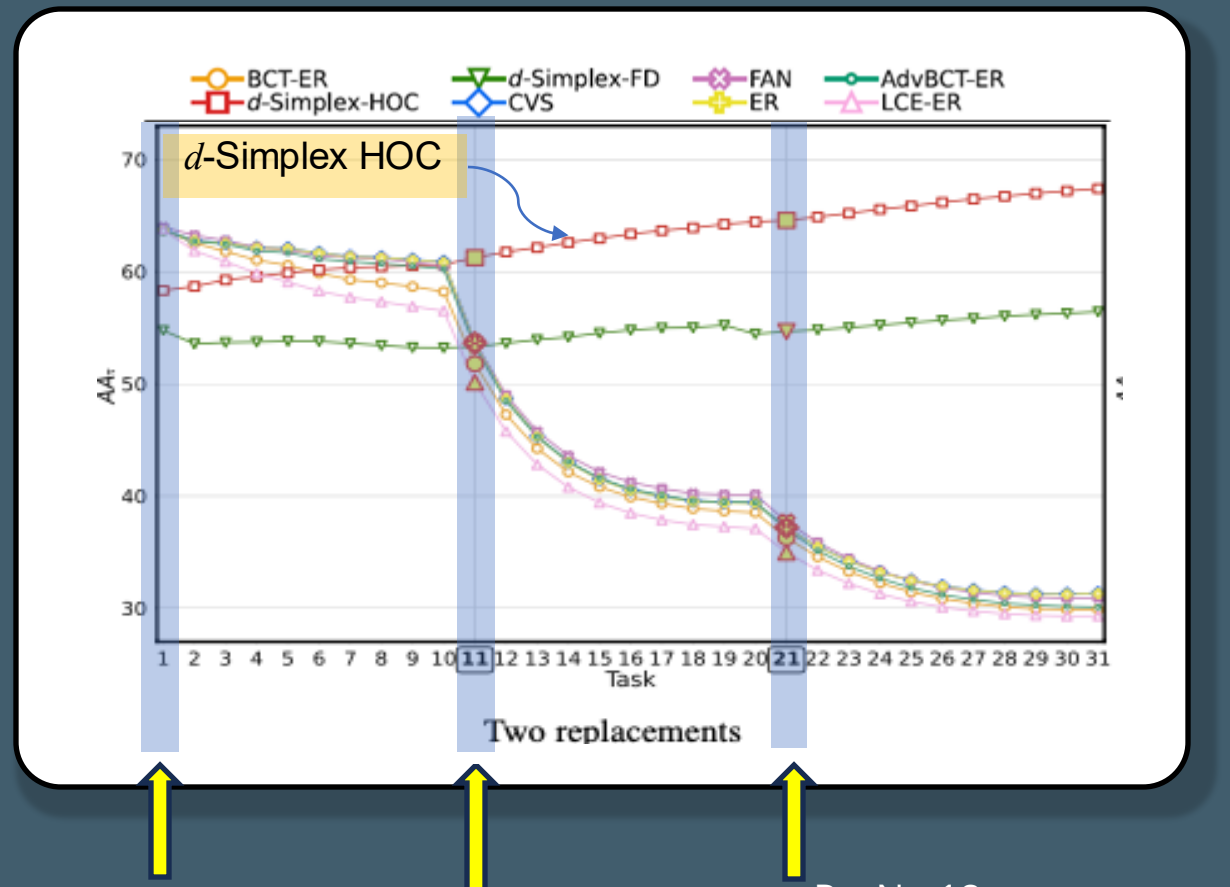
The now-standard practice is downloading and fine-tuning representations from large pre-trained models

Compatibility at the user-side should allow **seamless replacement** with the new improved versions of the large model



Updating with Replacement [*]

CIFAR10 dataset



d-Simplex fixed classifier with class pre-allocation provides **higher compatibility**

Fine-tuning CIFAR100R

ResNet18
pre-trained
100 ImageNet32 classes

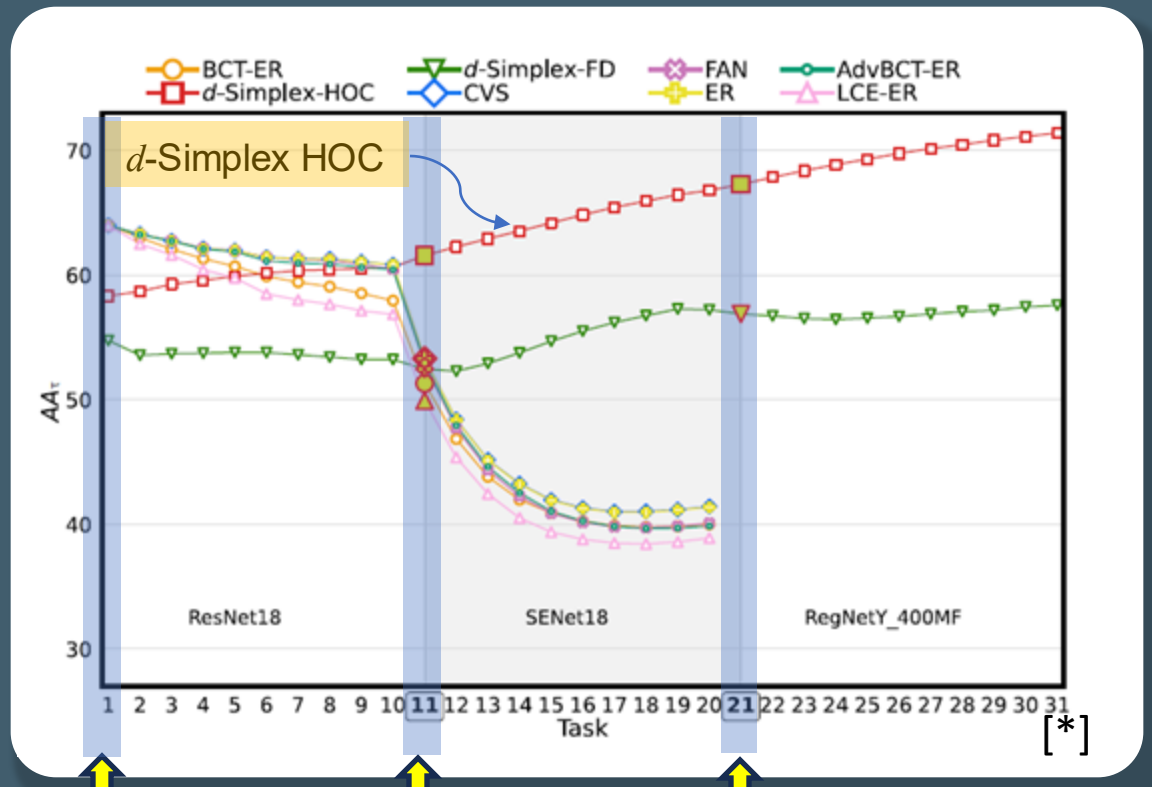
ResNet18
trained
300 ImageNet32 classes

ResNet18
trained
600 ImageNet32 classes

[*] N. Biondi et al., CVPR 2024

Updating with Replacement [*]

CIFAR10 dataset



Fine-tuning
CIFAR100R

ResNet18
pre-trained
100 ImageNet32 classes

Senet18
trained
300 ImageNet32 classes

RegNetY_400MF
trained
600 ImageNet32 classes

d-Simplex fixed classifier with class pre-allocation provides **higher compatibility** with **seamless model replacement**

The **fixed matrix** of the *d*-Simplex acts as a shared interface between the pre-trained model and the fine-tuned user model

[*] N. Biondi et al., CVPR 2024

Looking at the code

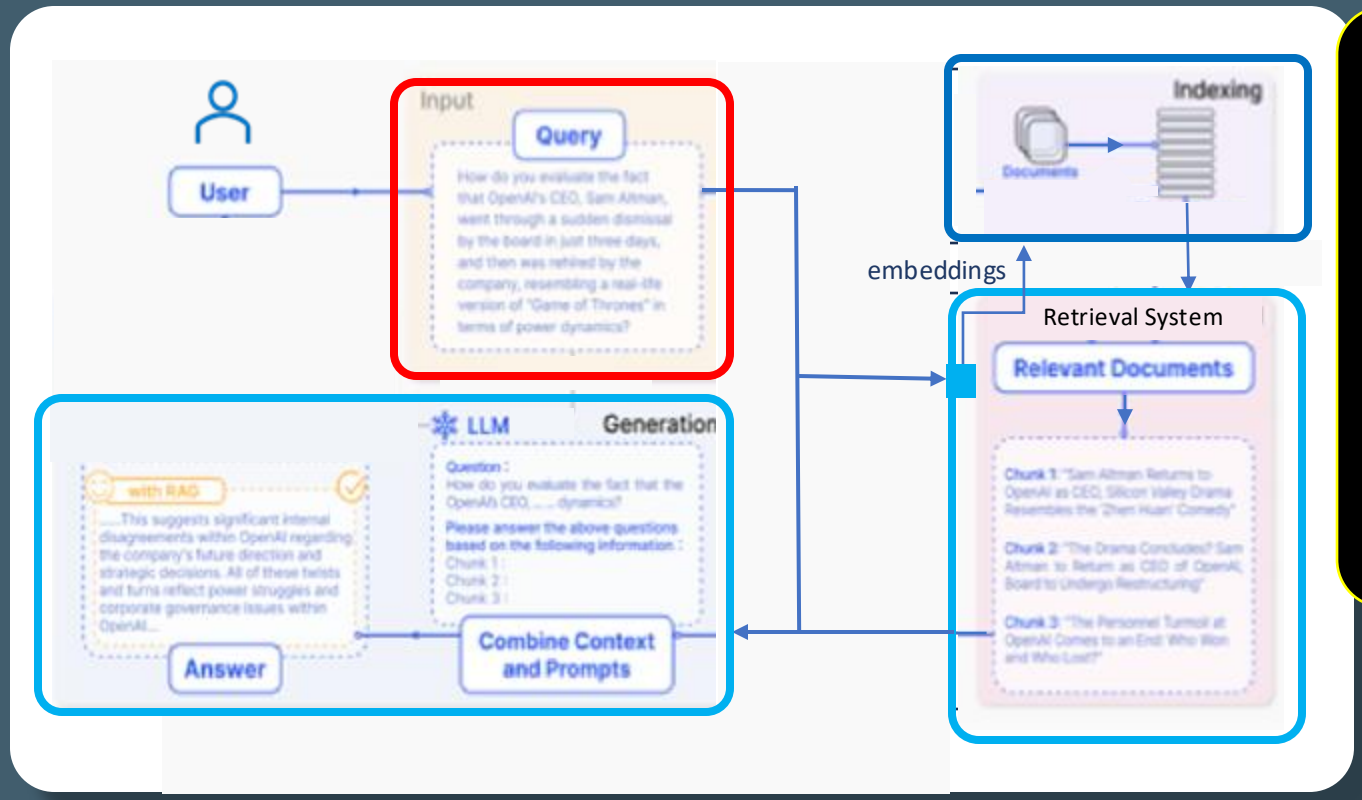
IAM-CL^{2}R Scenario

Google Colab link

Jupyter Notebook link

Updating With RAG

RAG: a promising solution by **incorporating knowledge from external databases** to enhance the accuracy and credibility of the generation

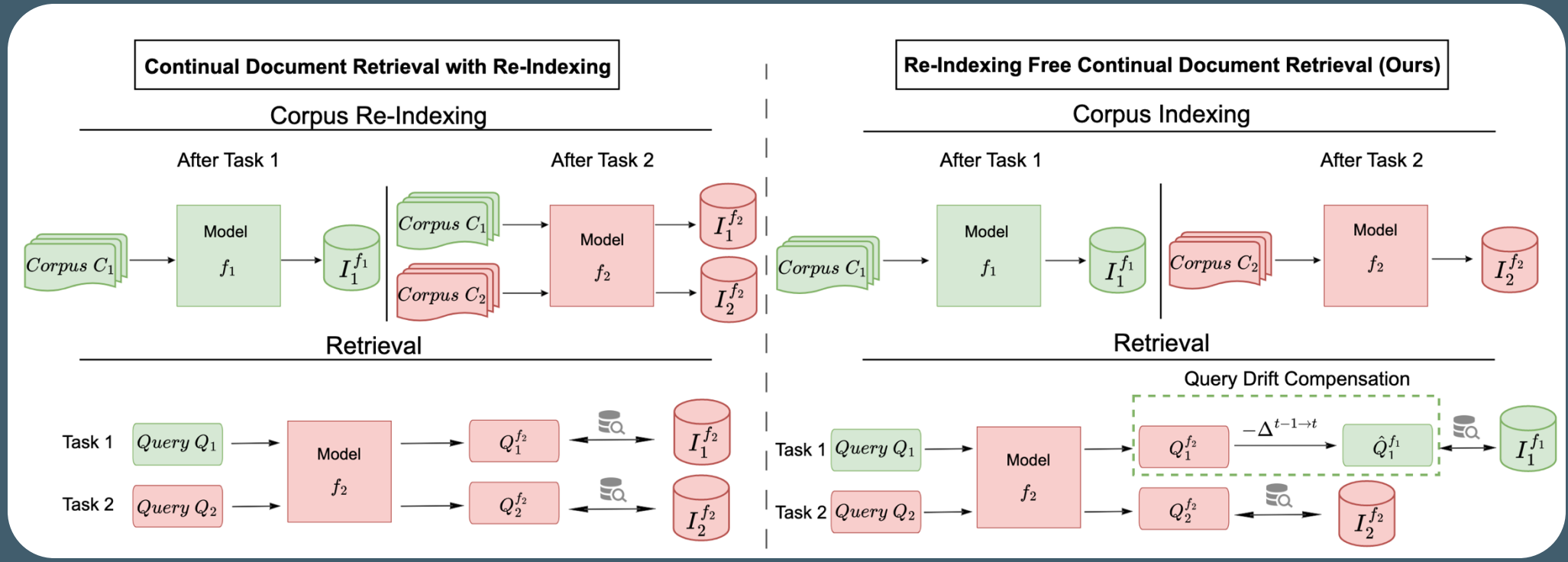


3 steps:

1) **Indexing** Documents are split into chunks, encoded into vectors, and stored in a database
2) **Retrieval** Retrieve the top k most relevant chunks based on semantic similarity.
3) **Generation** Input the original question and the retrieved chunks to generate the final answer

Compatibility with RAG

Compatibility **avoids re-indexing the vector database** when the large model is updated

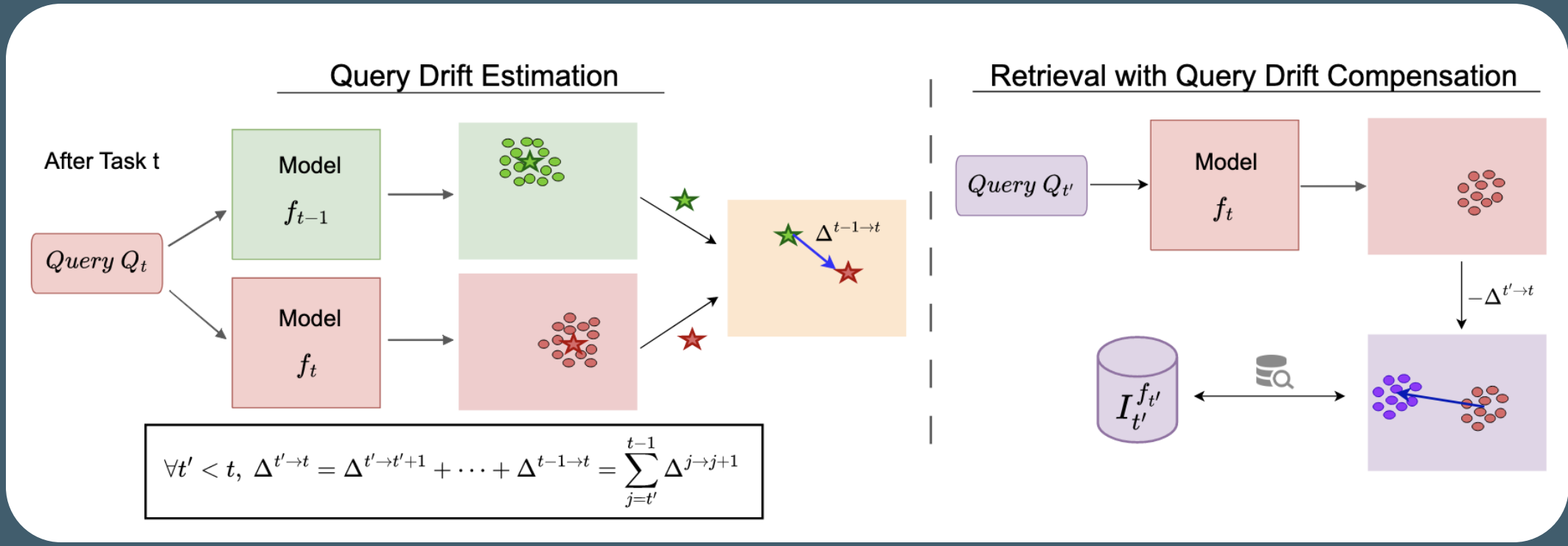


Picture authored by D. Goswami

Query Drift Compensation: Enabling Compatibility In Continual Learning of Retrieval Embedding Models[*]

QDC address compatibility in RAG systems by estimating **query drift vectors** for each task transition using the query samples from training data of the current task which are then used to approximate the query drift of previous tasks.

- ✓ These drift vectors are then subtracted from new model query embeddings to project them back to the old space.

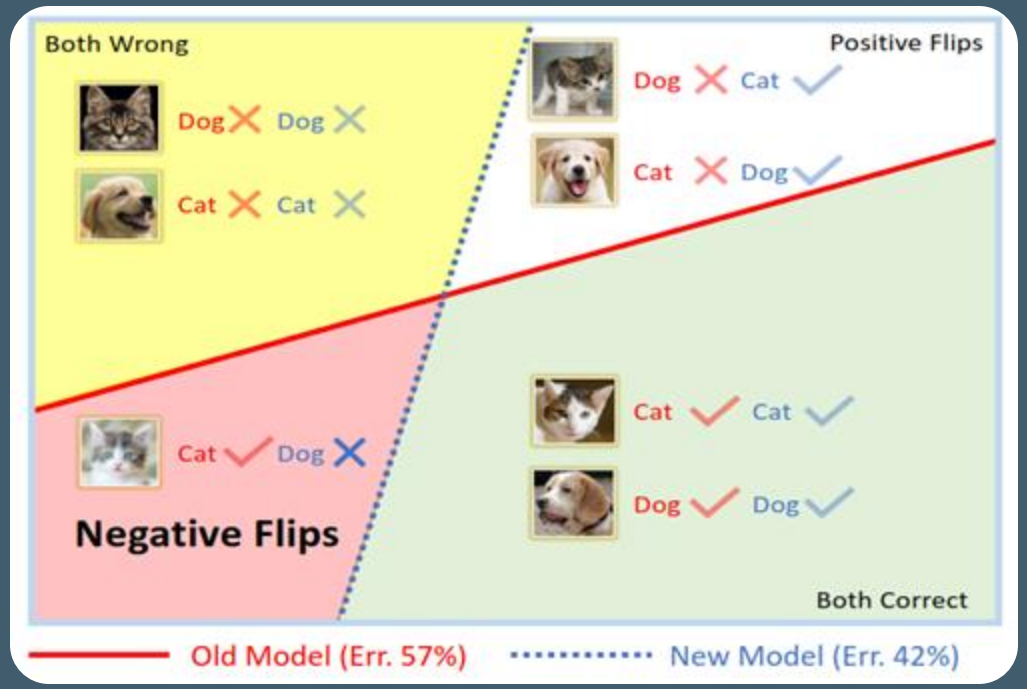


[*] Goswami et al., COLLAS 2025

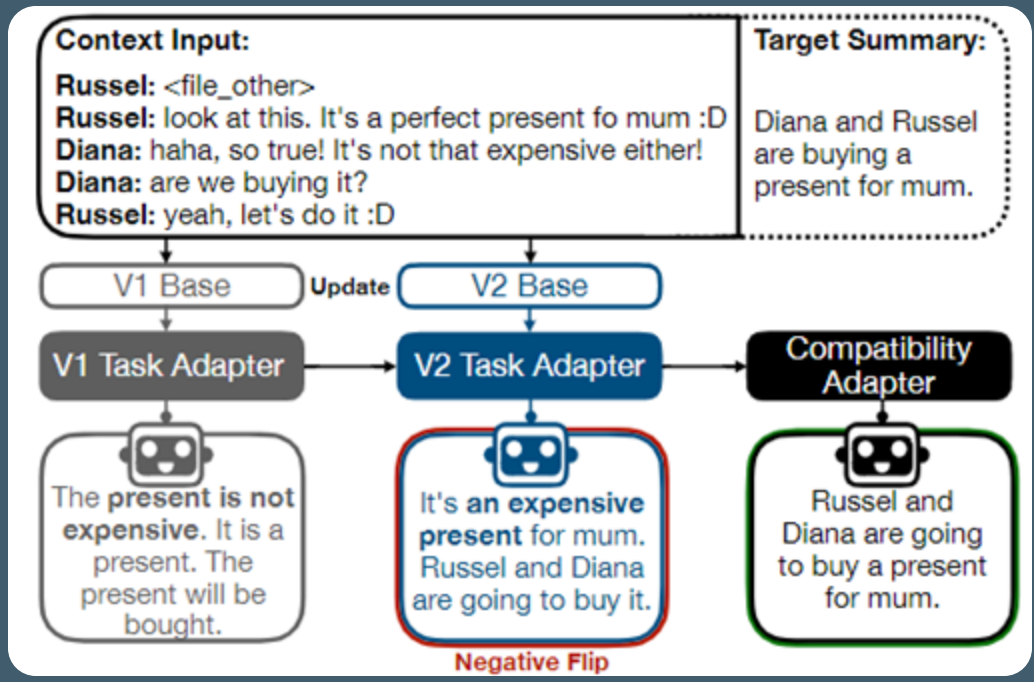
Third Party-side Compatibility

Upon updating the model should maintain consistency wrt the previous versions to minimize the disruption of the user experience.

Compatibility should aim at minimizing the number of **Negative Flips** i.e. previously correct instances that are predicted/generated incorrectly in the new version of the model



Discriminative task



Generative task

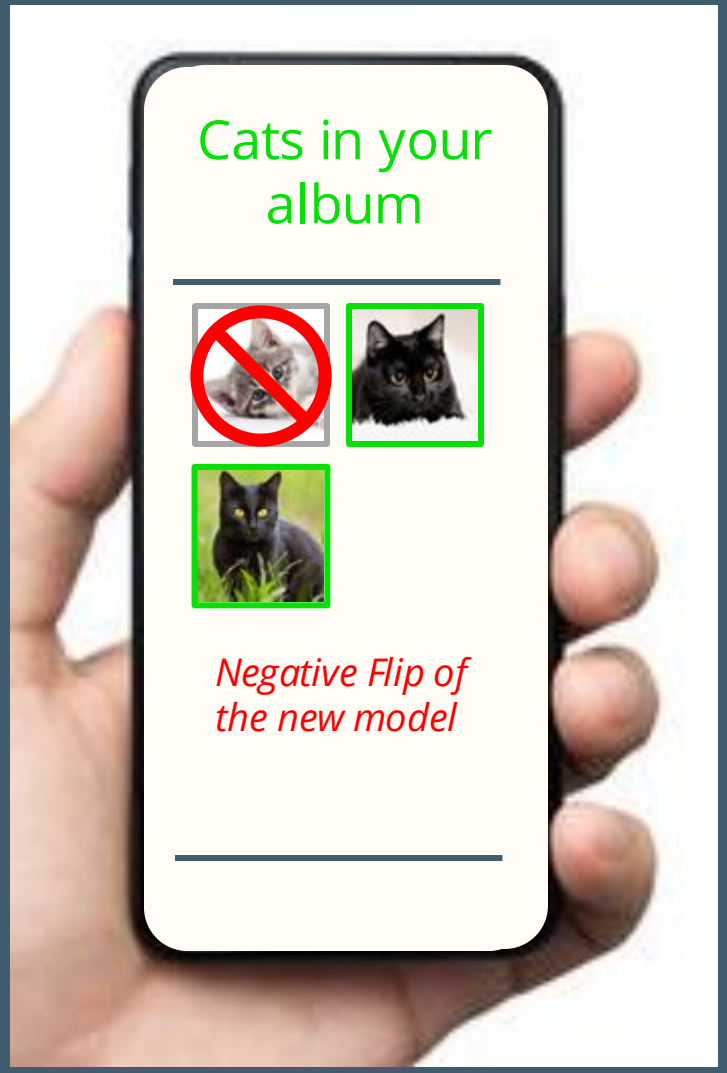
Negative Flips in Discriminative Tasks

The goal is minimizing both the **Error Rate** and the **Negative Flip Rate**

$$\mathrm{NFR} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}(\hat{y}_i^{\mathrm{new}} \neq y_i \wedge \hat{y}_i^{\mathrm{old}} = y_i),$$

The Relative NFR metric takes both element into consideration:

$$\text{Rel-NFR} = \frac{\mathrm{NFR}}{(1 - \mathrm{ER}_{\mathrm{old}}) \cdot \mathrm{ER}_{\mathrm{new}}},$$



Discriminative task

Positive Congruent Training [*]

Positive-Congruent training aims to **minimize both the Error Rate and the Negative Flip Rate** with two **separate losses.**

$$\min_{w} \underbrace{\mathcal{L}_{\text{CE}}\left(\phi^{\text{new}}, w\right)}_{\text{Error rate}} + \lambda \underbrace{\mathcal{L}_{\text{PC}}(\phi^{\text{new}}, w; \phi^{\text{old}})}_{\text{Negative flips}}$$

where

$$\lambda \mathcal{L}_{\text{PC}}(\phi^{\text{new}}, w; \phi^{\text{old}}) = \mathcal{F}(x_i)\mathcal{D}(\phi_w^{\text{new}}(x_i), q(x_i))$$

Approach	$\mathcal{F}$	$\mathcal{D}$	q
Naive Basline	$\mathbf{1}(\hat{y}^{\text{old}}(x_i) = y_i)$	Cross Entropy	$\vec{y}_i$
Focal Distillation - KL (FD-KL)	$\alpha + \beta \cdot \mathbf{1}(\hat{y}^{\text{old}}(x_i) = y_i)$	τ-scaled KL-Divergence	$p^{\text{old}}(y\|x_i)$
Focal Distillation - Logit Matching (FD-LM)	$\alpha + \beta \cdot \mathbf{1}(\hat{y}^{\text{old}}(x_i) = y_i)$	ℓ_2 distance	$\phi_w^{\text{old}}(x_i)$

The PC loss serves to bias training towards **maximal positive congruence** with old model

- ✓ F is a filter function that applies a weight for each training sample based on the **old model outputs**
- ✓ D is a distance function that measures the difference of the new model's outputs to a target vector $q(x_i)$

[*] S. Yan et al., CVPR 2021

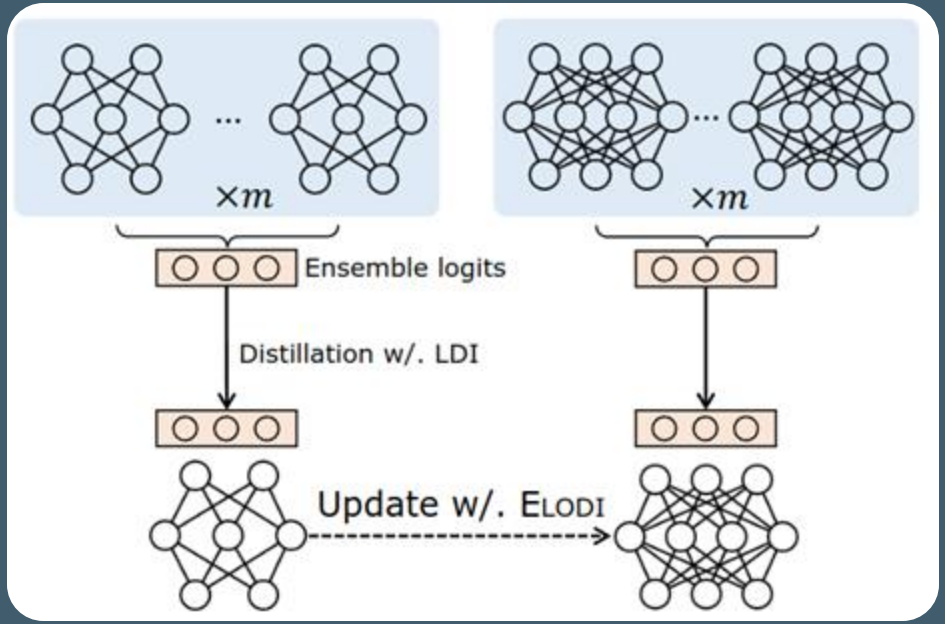
ELODI: Ensemble Logit Difference Inhibition for Positive-Congruent Training[*]

Deep **ensembles reduce NFR** by remedying potential flip samples that have large variations in the logits space of different single models

To reduce the cost of model ensembles the Logit Difference Inhibition (LDI) distillation loss is introduced

- ✓ LDI penalizes the logit difference between the ensemble and a single model on a subset of classes with the highest logit values
- ✓ The subset of classes with higher logit values is more prone to prediction flipping.

$$\mathcal{L}_{\mathrm{LDI}}(x) = \sum_{k \in \mathbb{K}(x)} \left(\| \phi_k(x) - \phi^{(\mathrm{ens})}(x) \|_p \right)^p$$



[*] Y. Zhao et al., IEEE TPAMI 2024

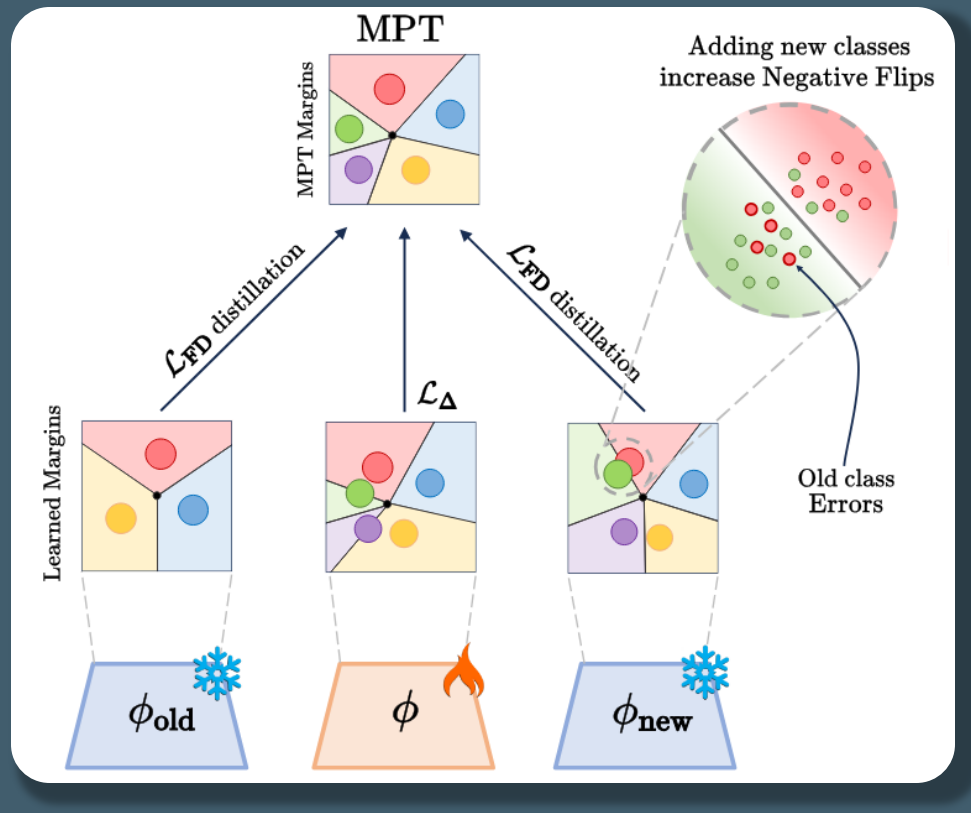
Mitigating Negative Flips via Margin Preserving Training[*]

Adding new categories **reduces the margin** of each class and may introduce conflicting patterns that undermine their learning process

Solution: induce a larger relative margin between the old and new classes by introducing an explicit margin-calibration term on the logits

$$\mathcal{L}_\Delta = -\log \frac{e^{\mathbf{z}_y + \Delta_y}}{\sum_{i=1}^{C} e^{\mathbf{z}_i + \Delta_i}}.$$

$$\Delta_i = \begin{cases} 0, & \text{if } i \in \mathcal{Y}^{\text{old}}, \\ k, & \text{if } i \in \mathcal{Y}^{\text{new}}. \end{cases}$$



Preserving the decision margins of old classes leads to underestimate those of new classes
=> distillation from a new model trained on all classes

[*] S. Ricci et al., Under Review

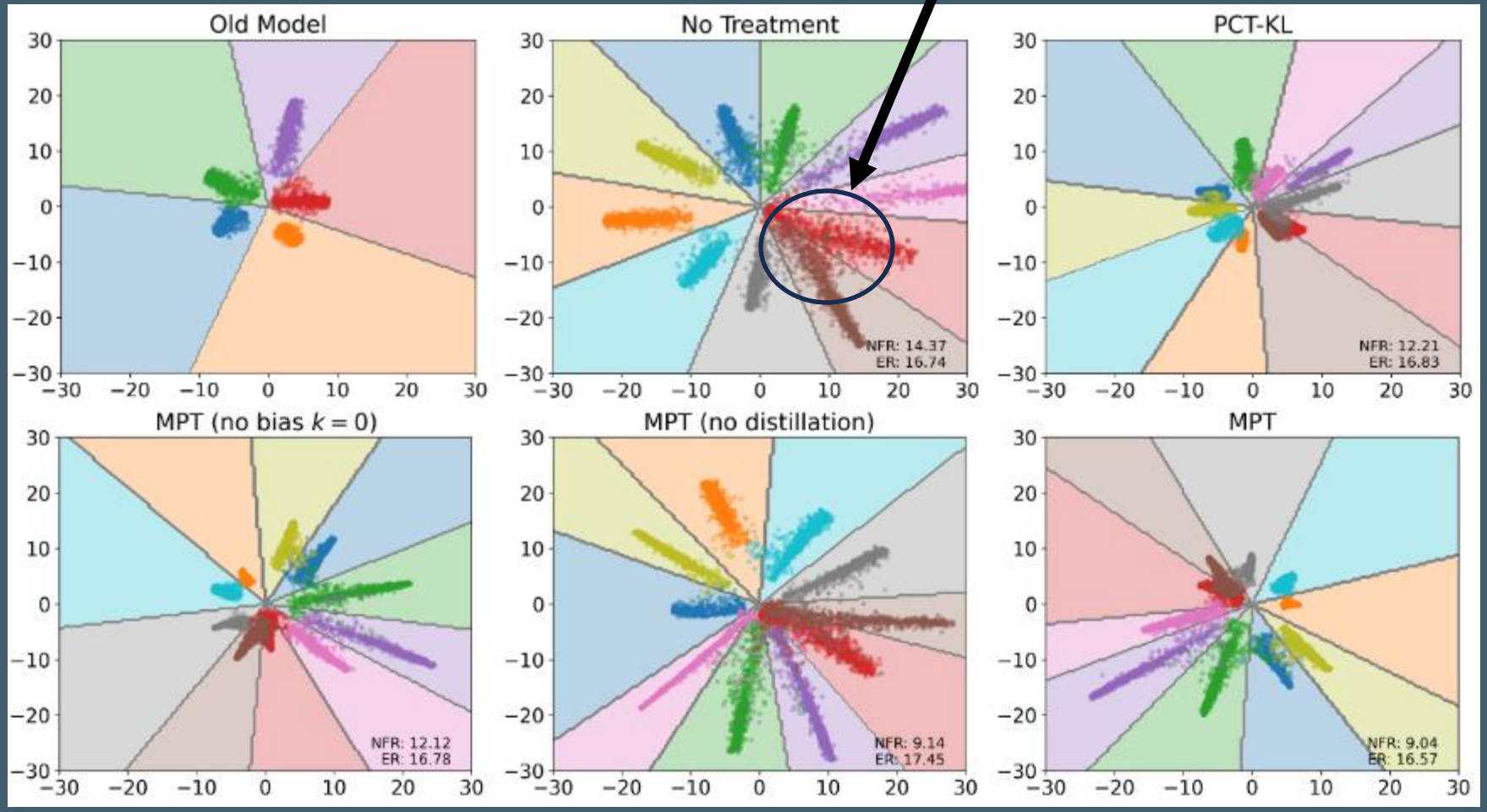
Negative Flip Reduction comparison of a pretrained ViT-B/32 fine-tuned on CIFAR100.

Method	$ER_\downarrow$(%) on $\mathcal{Y}^{old}$	$ER_\downarrow$(%) on $\mathcal{Y}^{all}$	$NFR_\downarrow$(%)	Rel-$NFR_\downarrow$(%)
Old model	13.28	-	-	-
No treatment	19.44	19.60	7.08	41.86
BCT (Shen et al. 2020b)	19.78	19.89	6.64	38.70
PCT-Naive (Yan et al. 2021)	17.40	19.73	5.26	34.85
PCT-KL (Yan et al. 2021)	19.44	19.52	6.88	40.81
PCT-LM (Yan et al. 2021)	19.50	19.59	6.58	38.91
ELODI (Zhao et al. 2024)	17.13	19.54	4.97	30.69
$ELODI_{TopK}$ (Zhao et al. 2024)	17.52	19.57	5.16	31.98
MPT-KL (Ours)	**15.16**	**19.51**	**3.26**	**24.09**
MPT-LM (Ours)	16.22	19.58	3.58	25.45

Negative Flip Reduction comparison of ResNet18 trained from scratch

Method	CIFAR100				ImageNet1k			
	$ER_\downarrow$(%) on $\mathcal{Y}^{old}$	$ER_\downarrow$(%) on $\mathcal{Y}^{all}$	$NFR_\downarrow$(%)	Rel-$NFR_\downarrow$(%)	$ER_\downarrow$(%) on $\mathcal{Y}^{old}$	$ER_\downarrow$(%) on $\mathcal{Y}^{all}$	$NFR_\downarrow$(%)	Rel-$NFR_\downarrow$(%)
Old model	33.27	-	-	-	25.55	-	-	-
No treatment	40.40	39.12	14.04	52.39	28.40	32.88	9.15	43.30
BCT (Shen et al. 2020b)	41.28	39.58	14.88	54.34	27.90	33.17	8.99	43.28
PCT-Naive (Yan et al. 2021)	40.54	40.32	14.04	52.20	26.22	33.17	7.24	37.06
PCT-KL (Yan et al. 2021)	39.44	38.79	11.94	45.63	27.73	32.69	8.09	39.19
PCT-LM (Yan et al. 2021)	41.34	41.83	12.12	44.19	27.98	34.70	7.55	36.24
ELODI (Zhao et al. 2024)	37.98	38.45	10.56	41.91	27.28	**32.13**	7.61	37.46
$ELODI_{TopK}$ (Zhao et al. 2024)	39.06	**38.32**	12.46	48.09	28.51	32.97	9.07	42.74
MPT-KL (Ours)	**34.32**	38.55	**9.26**	40.67	**24.68**	32.76	6.09	33.15
MPT-LM (Ours)	35.28	38.61	**9.26**	**39.57**	25.46	34.12	**6.02**	**31.76**

Negative Flips between red (old) and brown (new) classes



PCT-KL does not reduce NFR for the red class.

MPT addresses both ER and NFR

Negative Flips in Generative Tasks [*]

The NFR definition has been recently extended to the case of generative tasks (e.g. translation or summarization) with LLMs

PFR: positive flips rate

NFR: negative flips rate

$$\widetilde{\text{PFR}} \triangleq \frac{1}{N}\sum_{i}^{N} \mathbf{1}[D(x_i) > 0]$$

$$\widetilde{\text{NFR}} \triangleq \frac{1}{N}\sum_{i}^{N} \mathbf{1}[D(x_i) < 0]$$

where $D(x_i) \triangleq S(\mathcal{M}_{v2}(x_i), y_i) - S(\mathcal{M}_{v1}(x_i), y_i)$

is the distance between the two model outputs with respect to the groundtruth according to a similarity metric *S (BERT Score, ROUGE, BLEU Score ...)*

[*] J. Echterhoff et al., 2024

MUSCLE: A Model Update Strategy for Compatible LLM Evolution [*]

Compatibility (in term of NFR reduction) is achieved by fine-tuning the **local adapter** of the new LLM with a distillation procedure:

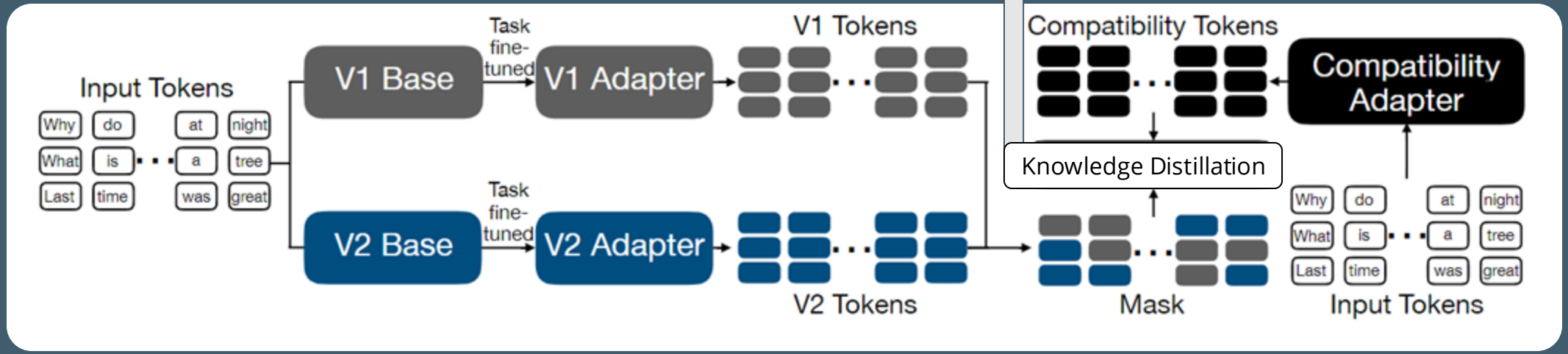
- ✓ if the compatible adapter predict the correct token it is aligned with the new model (improving performance)
- ✓ otherwise it is aligned with the old adapter to obtain consistency (reducing the negative flip)

$$m_i = \mathbf{1}[\text{argmax}\ \sigma(z_{\mathcal{M}^C_{v2},i}) \neq y_i]$$

$$a_{\mathcal{M}_{v1}} = KL(\sigma(z_{\mathcal{M}_{v1},i}/T) \| \sigma(z_{\mathcal{M}^C_{v2},i}/T))$$

$$a_{\mathcal{M}_{v2}} = KL(\sigma(z_{\mathcal{M}_{v2},i}/T) \| \sigma(z_{\mathcal{M}^C_{v2},i}/T))$$

$$\mathcal{L}^m_{comp} = \frac{1}{n}\sum_{i=1}^{n} m_i \cdot a_{\mathcal{M}_{v1}} + (1 - m_i) \cdot a_{\mathcal{M}_{v2}}$$



[*] J. Echterhoff et al., 2024

Thanks for your attention!

Check it out!
Repo with slide and code!

Niccolò Biondi

Simone Ricci

Alberto Del Bimbo

github.com/NiccoBiondi/compatibility-tutorial

References

N. Biondi et al., *Cl2r: Compatible lifelong learning representations*, ACM TOMM2023

N. Biondi et al., *Cores: Compatible representations via stationarity*, IEEE TPAMI2023

N. Biondi et al., *Stationary representations: Optimally approximating compatibility and implications for improved model replacements*, CVPR2024

N. Biondi et al., *Stationary (and therefore compatible) representation is all you need,* IEEE TPAMI 2024 (submitted)

R. Bommasani et al. , *On the opportunities and risks of foundation models*, ArXiv2022

K. Chen et al., *R3 adversarial network for cross model face recognition*, CVPR2019

M. de Lange et al.., *A continual learning survey: defining forgetting in classification tasks*, IEEE TPAMI 2021

J. Echterhof et al., *MUSCLE: A Model Update Strategy for Compatible LLM Evolution,* 2024

E. Hoffer et al., *Fix your classifier: the marginal value of training the last weight layer*, ICLR2018

A. Iscen et al., *Memory-efficient incremental learning through feature adaptation*, CVPR2020

Z. Li & D. Hoiem, *Learning without forgetting*, IEEE TPAMI2017

D. Mc Candless, blog: *https://informationisbeautiful.net/visualizations/the-rise-of-generative-ai-large-language-models-llms-like-chatgpt*

Q. Meng et al., *Learning compatible embeddings*, ICCV2021

References

R. Merrit, *What is a transformer model?*, NVIDIA blog

F. Pernici et al., *Regular polytope networks*, TNNLS2021

F. Pernici et al., *Class-incremental learning with pre-allocated fixed classifiers*, ICPR2020

F. Pernici et al., *Maximally Compact and Separated Features with Regular Polytope Networks*, CVPRW2019

A. Radford et al., *Learning transferable visual models from natural language supervision,* ICML2021

S. A Rebuffi et al., *iCARL incremental classifier and representation learning*, CVPR2017

Y. Shen et al. *Towards backward-compatible representation learning*, CVPR2020

G. van de Ven et al., *Three types of incremental learning*, Nature Machine Intelligence 2022

A. Vaswani et al., *Attention is all you need,* NeurIPS2017

CY. Wang et al., *Unified Representation Learning for Cross Model Compatibility*, BMVC2020

TST. Wan et al., *Continual learning for visual search with backward consistent feature embedding*, CVPR2022

B. Zhang et al., *Towards Universal Backward-Compatible Representation Learning*, IJCAI 2022

B. Zhao et al., *Continual representation learning for biometric identification*, WACV 2021

Y. Zhao et al., *ELODI: Ensemble Logit Difference Inhibition for Positive-Congruent Training*, IEEE TPAMI2024