Few-Shots Class-Incremental-Learning for Face Recognition

Aryan Tomar (2003107)

BTech Presentation

Project Instructor: Dr. Shitala Prasad
Indian Institute of Technology Goa

Problem Statement

Our aim was to build a Few-Shots Class-Incremental-Learning Face Recognition model. We experimented different approaches suitable for the method and provided theoretical reasoning for all our claims while formulating the adequate model suitable for the task. The training and testing strategy introduced in the work are our novelty. We used a combination of existing loss functions in a novel way and formulated an overall loss function which is also our contribution.

TABLE OF CONTENTS

Phase 0

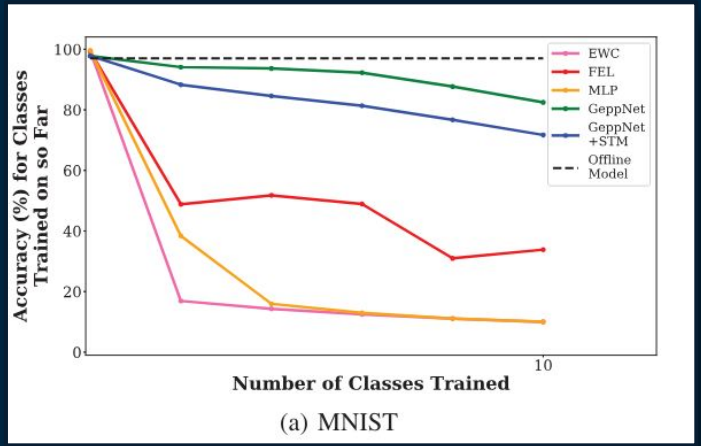
What's the need for Continual Learning?

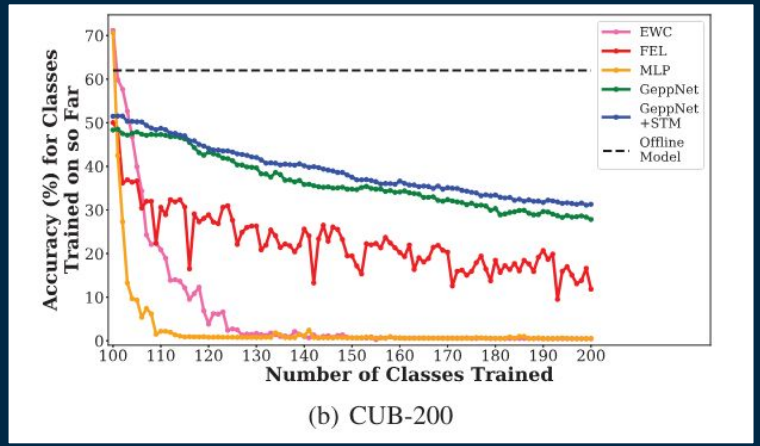
- **50GB/s** streaming data.
- **~30240 TB of data** after only a week.
- **Impossible** to re-train the mini-spot brain from scratch.

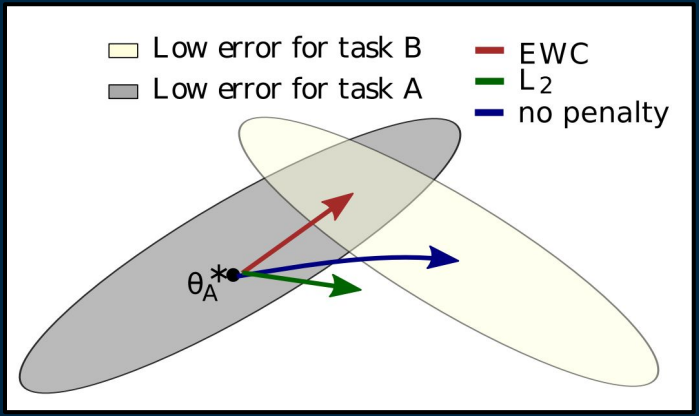
What's Catastrophic Forgetting?

Forgetting in Classical Neural Networks (MLP)

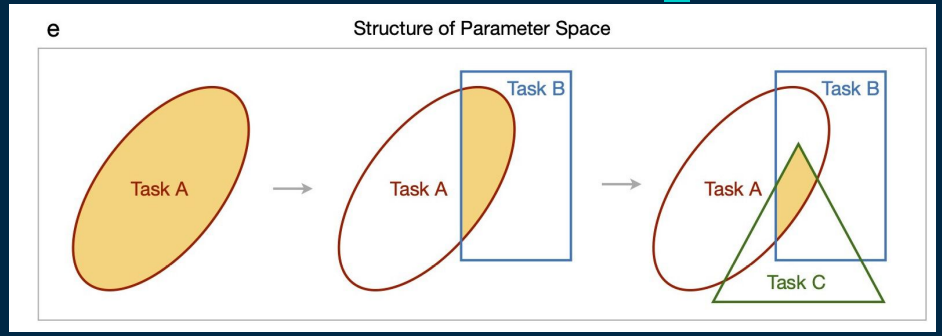
*Catastrophic interference, also known as **catastrophic forgetting**, is the tendency of an artificial neural networks to completely and abruptly forget previously learned information upon learning new information. ->* Mostly due to Gradient Descent.







1. *The network can't access all the classes at once, rather the data comes in an incremental fashion.*
2. *Model is evaluated over all the seen classes.*
3. *Same model is trained in each incremental step.*



Definition 1. Class-Incremental Learning *aims to learn from an evolutive stream with incoming new classes [32]. Assume there is a sequence of B training tasks[1] $\{\mathcal{D}^1, \mathcal{D}^2, \cdots, \mathcal{D}^B\}$ without overlapping classes, where $\mathcal{D}^b = \{(\mathbf{x}_i^b, y_i^b)\}_{i=1}^{n_b}$ is the b-th incremental step with n_b training instances. $\mathbf{x}_i^b \in \mathbb{R}^D$ is an instance of class $y_i \in Y_b$, Y_b is the label space of task b, where $Y_b \cap Y_{b'} = \varnothing$ for $b \neq b'$. We can only access data from $\mathcal{D}^b$ when training task b. The ultimate goal of CIL is to continually build a classification model for all classes. In other words, the model should not only acquire the knowledge from the current task $\mathcal{D}^b$ but also preserve the knowledge from former tasks. After each task, the trained model is evaluated over all seen classes $\mathcal{Y}_b = Y_1 \cup \cdots Y_b$.*

How to Solve Forgetting?

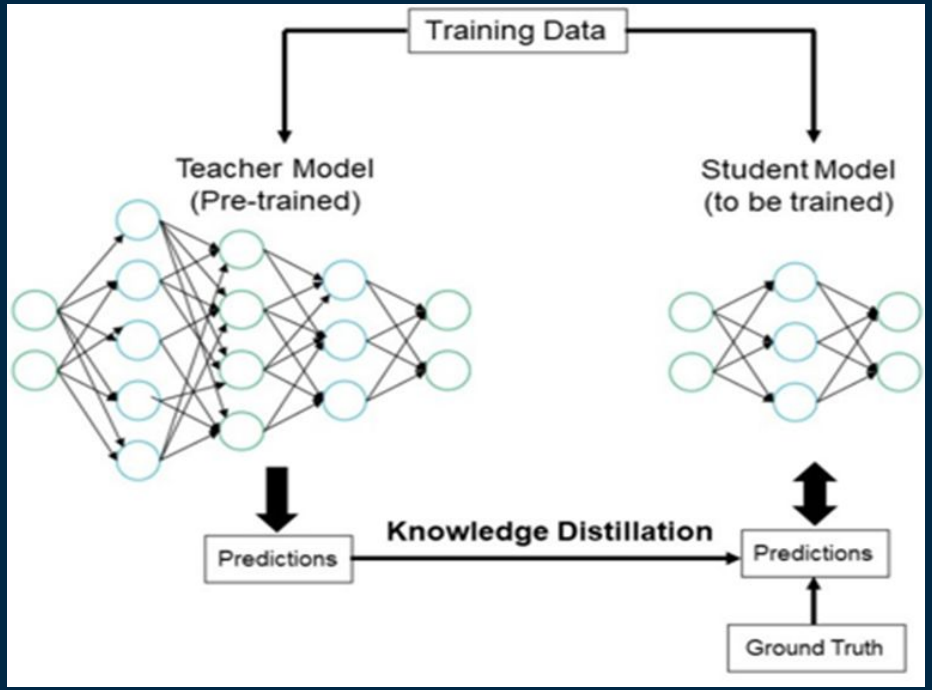
- Can we store just a portion of the previously encountered data?
- Can we make an ensemble of models?
- **Can we freeze some parameters of the network?**
- Have the model parameters the same importance? Can we leverage this to avoid forgetting?

Algorithm Category	Subcategory	
§ 3.1 Data-Centric Class-Incremental Learning	§ 3.1.1 Data Replay	Direct Replay
		Generative Replay
	§ 3.1.2: Data Regularization	
§ 3.2 Model-Centric Class-Incremental Learning	§ 3.2.1 Dynamic Networks	Neuron Expansion
		Backbone Expansion
		Prompt Expansion
	§ 3.2.2: Parameter Regularization	
§ 3.3 Algorithm-Centric Class-Incremental Learning	§ 3.3.1 Knowledge Distillation	Logit Distillation
		Feature Distillation
		Relational Distillation
	§ 3.3.2 Model Rectify	Feature Rectify
		Logit Rectify
		Weight Rectify

Phase 1

Vanilla CIL algorithm (using Knowledge Distillation)

- Distilling the old model knowledge while training the model for new classes.
- Core idea of knowledge distillation-based CIL methods is to build the mapping between old and new models and reflect the characteristics of the old model in the updating process.
- KL-Divergence is used as the knowledge distiller in our model.



Developing our Baseline Model

- Classification loss: Cross-Entropy
- Face Recognition loss: Triplet Loss learns robust representations separating the dissimilar classes and bringing sam classes together.
- Knowledge Distillation: KL-Divergence between the old and the new model. τ is the temperature and set to 2 (ref. Hinton et al).
- Vanilla CIL algorithm for face recognition system would be the combination of the above loss functions.

$$\frac{1}{B}\sum_{i=1}^{B}\mathcal{L}_{CE}\left(y_i, \sigma(f(x_i))\right)$$

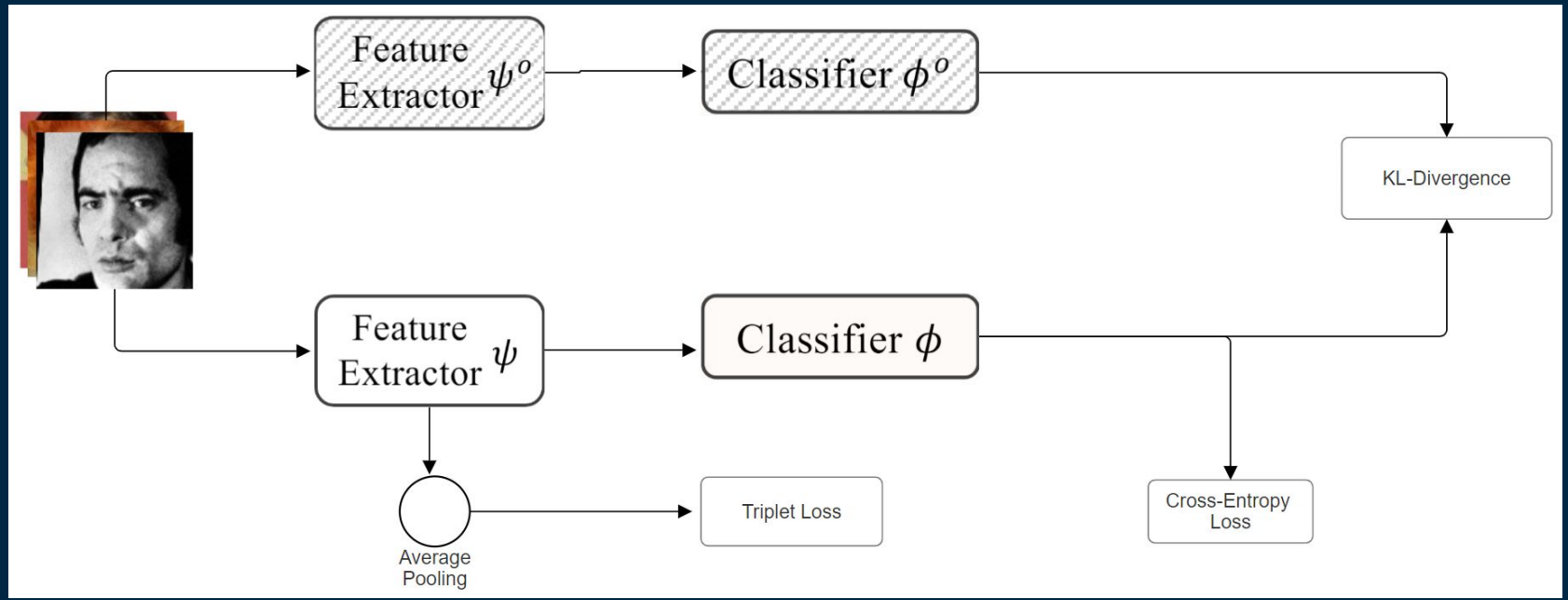
$$+\frac{1}{B}\sum_{i=1}^{B}\mathcal{L}_{tri}\left(g(\psi(x_i)), g(\psi(x_i^p)), g(\psi(x_i^n))\right),$$

$$\mathcal{L}_{KD} = \frac{1}{B}\sum_{i=1}^{B}\mathrm{KL}\left(\sigma(f^o(x_i)/\tau) \,\|\, \sigma(f(x_i)/\tau)\right),$$

where τ is a hyperparameter referred to as the temperature

$$\mathcal{L}_{base} = \mathcal{L}_{CE} + \mathcal{L}_{triplet} + \mathcal{L}_{KD}$$

Baseline Model Architecture



$$\mathcal{L}_{base} = \mathcal{L}_{CE} + \mathcal{L}_{triplet} + \mathcal{L}_{KD}$$

A New Training Strategy (Novelty)

The existing CIL methods for Face Recognition revolves around the following training strategy: -

- *Incremental Classes are taken only from one dataset by splitting it up into 5/10/20 incremental steps containing equal number of classes.*
- *Each class is trained on 100s of number of samples for that class.*

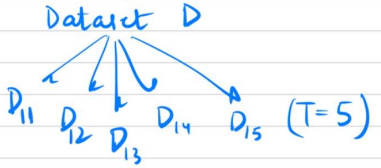
*To make our Face Recognition model **more suitable for real world settings**, we take: -*

- *Multiple (Q=5) face recognition datasets (for variations in images, since images taken in each dataset has its own challenges to learn) as our incremental steps, with one dataset being equivalent to one incremental step.*
- *Only train on K=4 samples per class, making it a few-shots learning problem. [Motivation: Human Learning capability]*

A New Training Strategy (Novelty) Explained in detail

Existing

Dataset D



1. $M_0 \xrightarrow{D_{11}} M_1$
2. $M_1 \xrightarrow{D_{12}} M_2$
5. $M_4 \xrightarrow{D_{15}} M_5$

Same no. of classes in $D_1 \cdots D_5$

$D_{11} \rightarrow \{c_1, c_2, \cdots c_n\}$

Each class c_i contains ≥ 100 samples

New Strategy

5 Datasets → D_1 D_2 D_3 D_4 D_5
(d=5)

Number of classes for a dataset are same
= Least no. of classes $\{D_1, \cdots, D_5\}$

1. $M_0 \xrightarrow{D_1} M_1$
2. $M_1 \xrightarrow{D_2} M_2$
5. $M_4 \xrightarrow{D_5} M_5$

$D_1 \rightarrow \{c_1, c_2, \cdots c_n\}$

Each c_i contains K=4 samples

Dataset Used

UMDFace

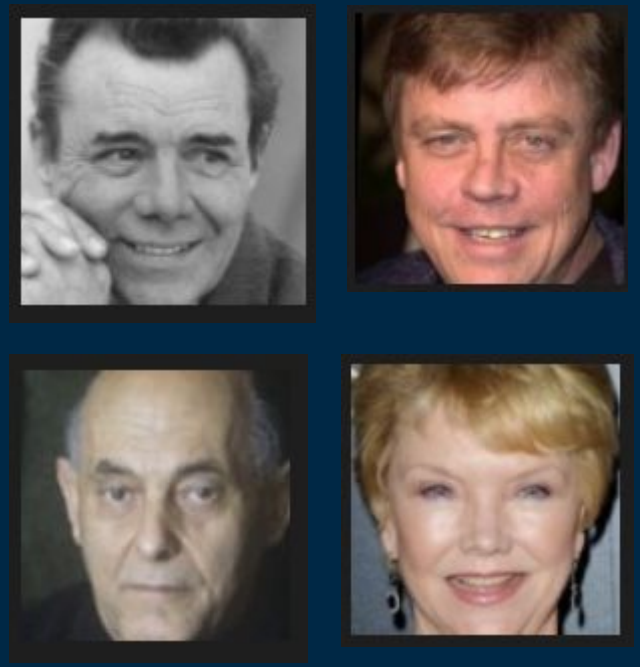
- 367,888 face annotations for 8,277 subjects
- Released in 2016 and maintained by University of Maryland, College Park.
- Challenges include variation in pose and expression as seen in each column of the image ->

Figure 1. Few samples from the dataset discussed in the paper. Each column represents variations in pose and expression of images of a subject.

Dataset Used

ArcFace (MS1Mv2)

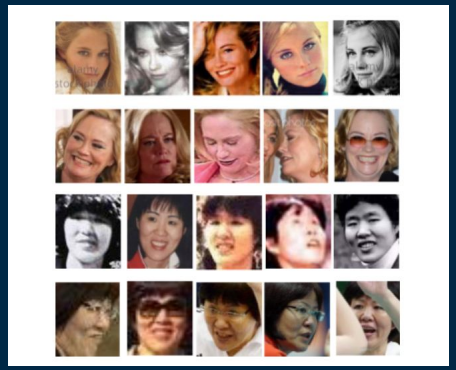
- 5.8 Millions images of 85K identities.
- MS1Mv2 is a semi-automatic refined version of the MS-Celeb-1M dataset.
- Employed ethnicity-specific annotators for large-scale face image annotations for refinement.
- Relatively, less challenging due to the refinement process.



Dataset Used

VGGFace2

- 3.31 million images divided into 9131 classes.
- Created and maintained by Visual Geometry Group, Oxford.
- Most challenging dataset, due to the variations in a single class. (360 imgs/class on average). Variations mostly include age variations and pose variations.



Age variations

Pose Variations

Dataset Used

RetinaFace

- 5.1 Millions images of 93K identities.
- RetinaFace is a semi-automatic refined version of the MS-Celeb-1M dataset using pre-trained ArcFace.
- Employed ethnicity-specific annotators.
- Even more refined dataset than ArcFace. The refinement process involved facial landmark detection and re-alignment of the image.

Dataset Used

CasiaFace

- The dataset contains 494,414 face images of 10,575 real identities.
- Dataset is created from web-scraping, and no data pre-processing, so very war images are present.
- Challenge: contains low resolution images and occluded images.

Training Settings

- ResNet-50 as Backbone
- Training on 5 incremental steps (5 datasets), one dataset at a time.
- At each step, we train on 50 epochs and 150 iterations per epoch using Adam optimizer.
- The learning rate is set to 3.5×10^{-4} initially and decays by x0.1 at 25th and 35th epochs.
- Each batch is composed of 32 identities and 4 images per identity. B=128.
- The input images are resized to 256 x 128 with data augmentations including random cropping, horizontal flipping and erasing.
- τ the temperature for KLD is set at 2.
- 8000 classes are randomly selected from each dataset.
- Training Sequence: UMDFace -> ArcFace -> VGGFace -> RetinaFace -> CASIA-WebFace

Evaluation criteria (top-1 accuracy)

- For a class, we take K=4 samples for training, one sample as query and rest of the samples in the gallery data. We then repeat this process for all the classes in the dataset. This gives a big gallery dataset to look for the image while testing, we then use top-1 accuracy to detect our test image in from the gallery dataset. We calculate the distance of a query image with all of the images in the gallery and report the top detection as 0 if not of same class and 1 if of same class.

Results (without KD)

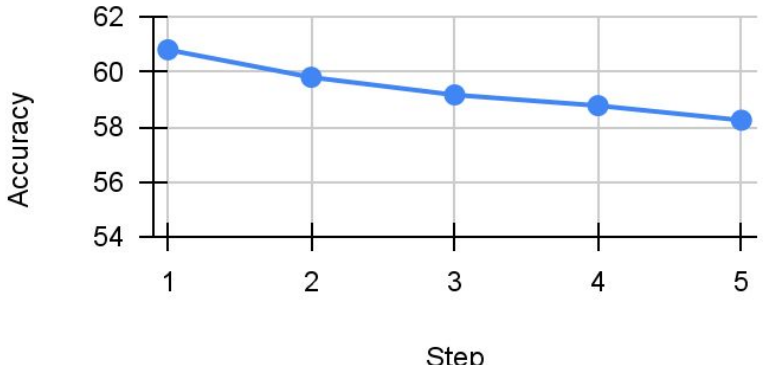
Test →

Train ↓

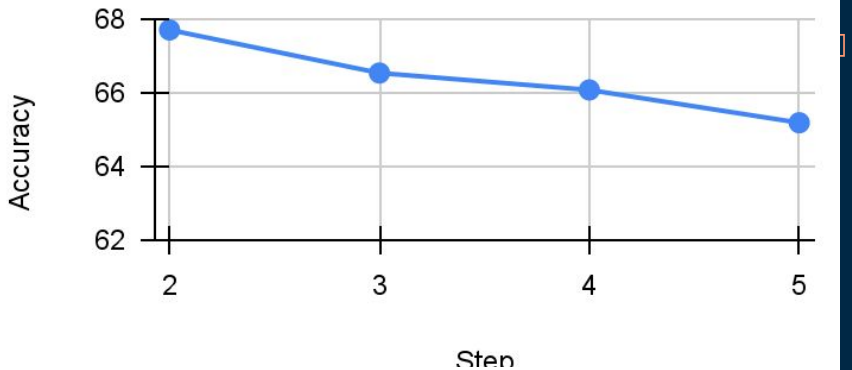
	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**60.8**				
ArcFace	**59.8**	**67.71**			
VGGFace	**59.16**	**66.54**	**47.22**		
RetinaFace	**58.77**	**66.08**	**46.15**	**61.86**	
CasiaFace	**58.24**	**65.19**	**45.69**	**59.27**	**42.26**
	-2.56	-2.52	-1.53	-2.59	

Results (without KD)

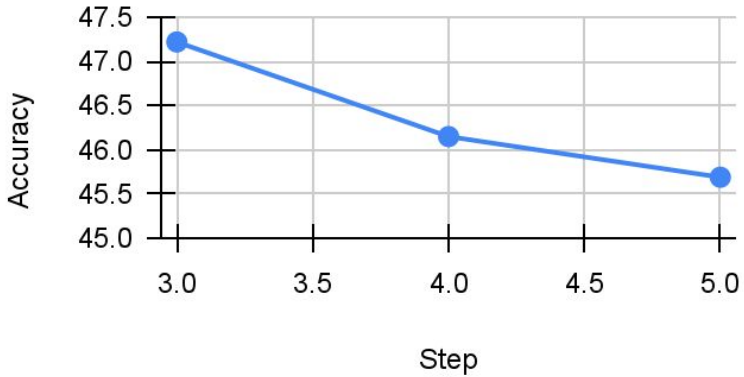
UMDFace (1)



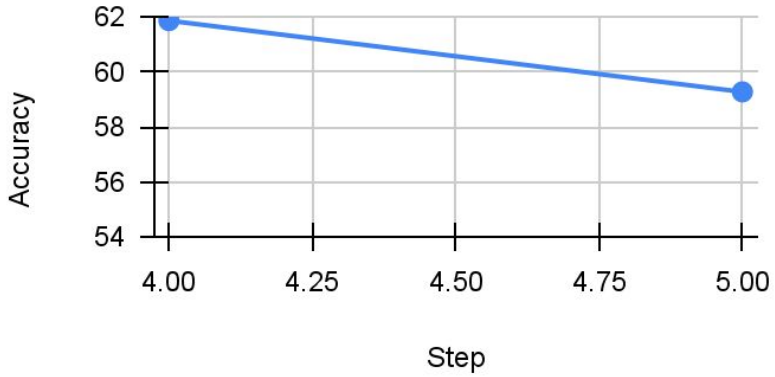
ArcFace (2)



VGGFace (3)



RetinaFace (4)



Results (with KD)

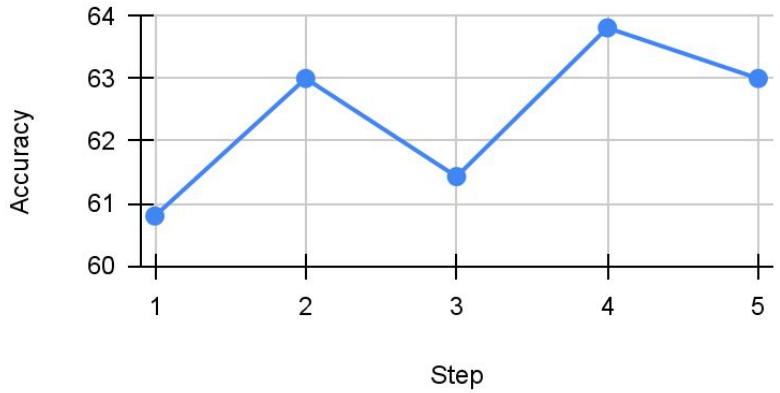
Test →

Train ↓

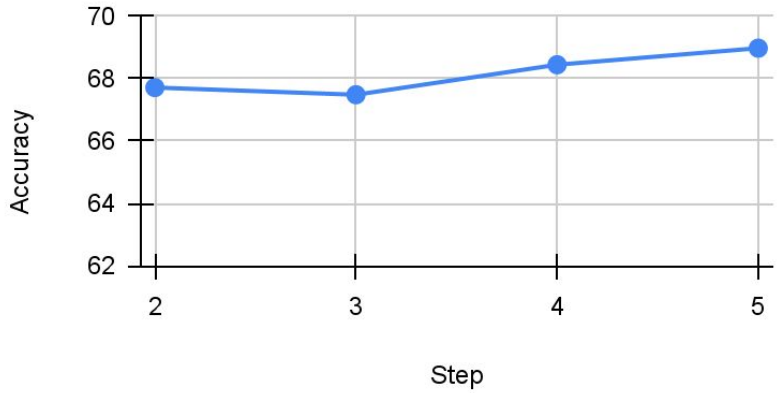
	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**60.8**				
ArcFace	**63**	**67.71**			
VGGFace	**61.43**	**67.48**	**47.27**		
RetinaFace	**63.81**	**68.44**	**48.44**	**61.98**	
CasiaFace	**63**	**68.97**	**49.46**	**62.28**	**42.2**
	+2.2	+1.26	+1.99	+0.3	

Results (with KD)

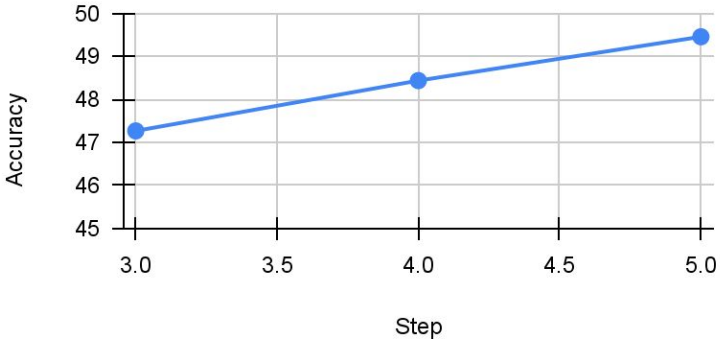
UMDFace (1)



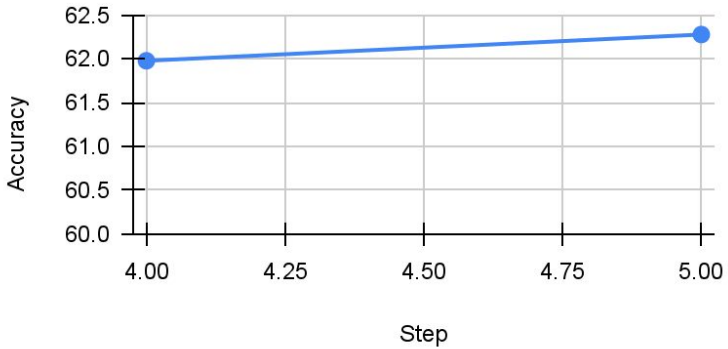
ArcFace (2)



VGGFace (3)



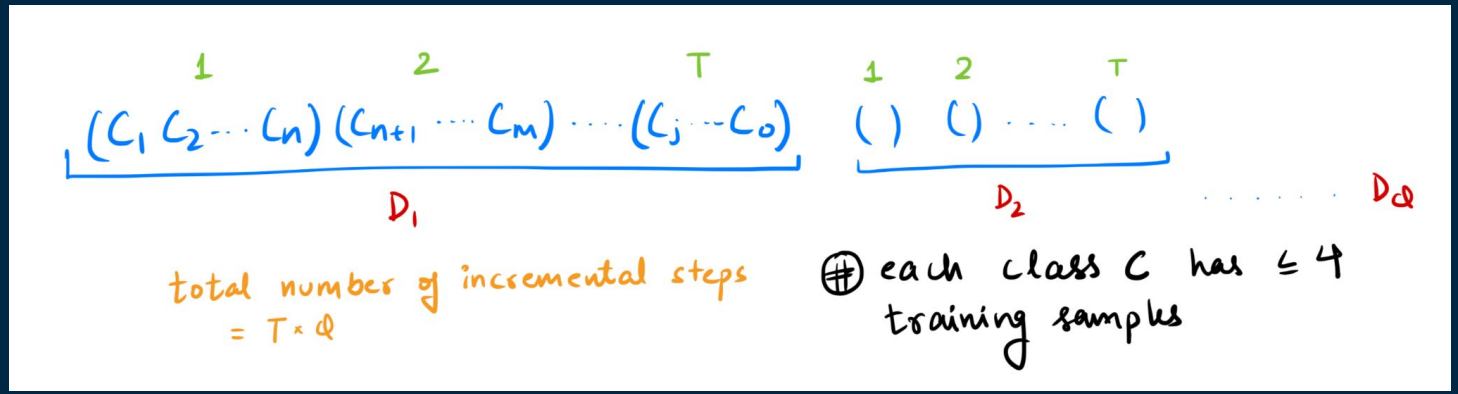
RetinaFace (4)



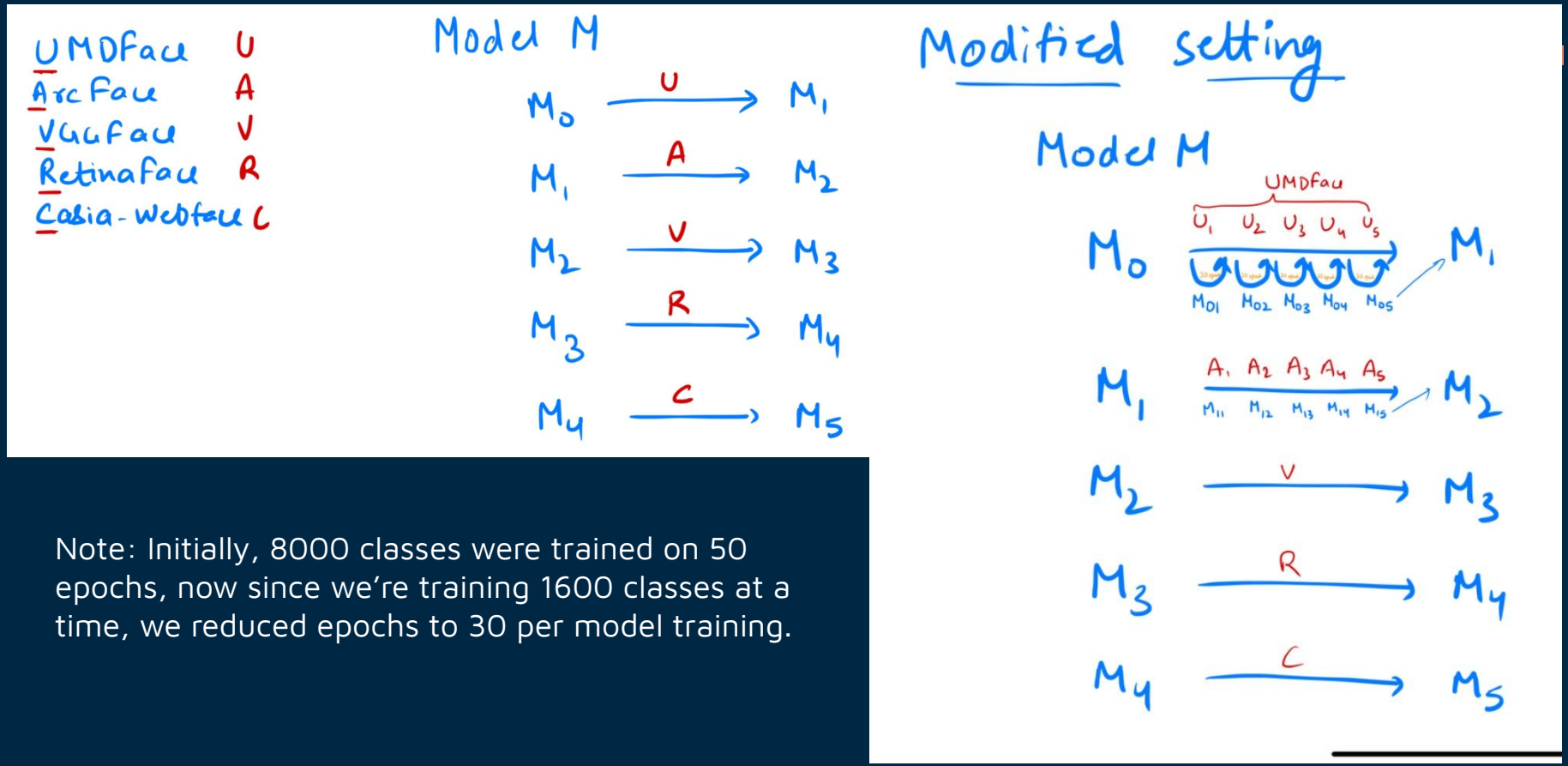
Phase 2

Training Strategy Modification (Novelty)

- To make our Face Recognition system more scalable, we increase the number of incremental states.
- Each dataset is split-up into T number of incremental steps, containing equal number of classes.
- Each class is trained with only K number of samples (K ≤ 4).



Training Strategy Explained in detail



Note: Initially, 8000 classes were trained on 50 epochs, now since we're training 1600 classes at a time, we reduced epochs to 30 per model training.

Results for T=5, K=4 (with KD)

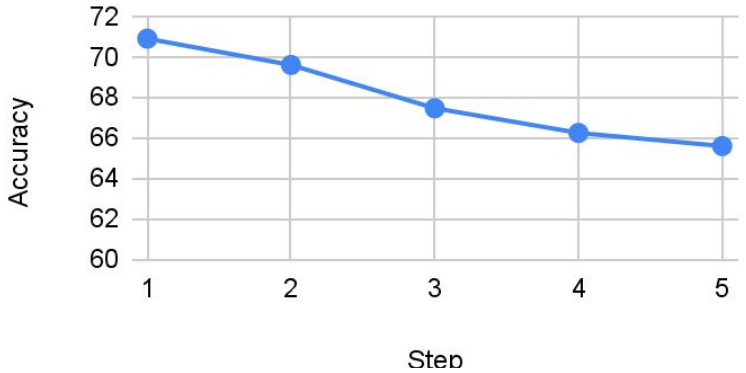
Test →

Train ↓

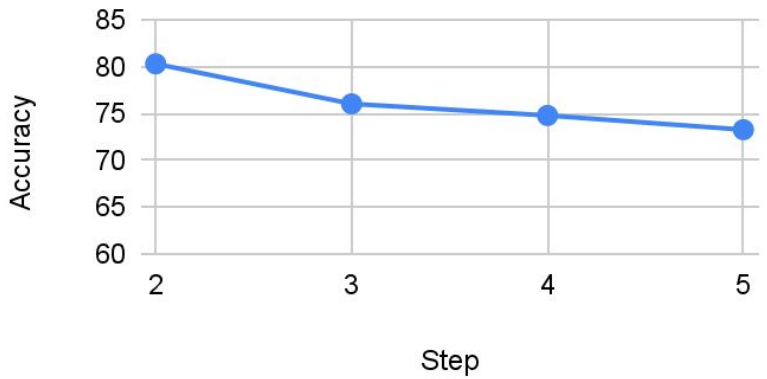
	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**70.89**				
ArcFace	**69.59**	**80.3**			
VGGFace	**67.45**	**76.03**	**61.23**		
RetinaFace	**66.23**	**74.79**	**58.13**	**72.14**	
CasiaFace	**65.58**	**73.27**	**55.36**	**68.47**	**58.72**
	-5.31	-7.03	-5.87	-3.67	

Results for T=5, K=4 (with KD)

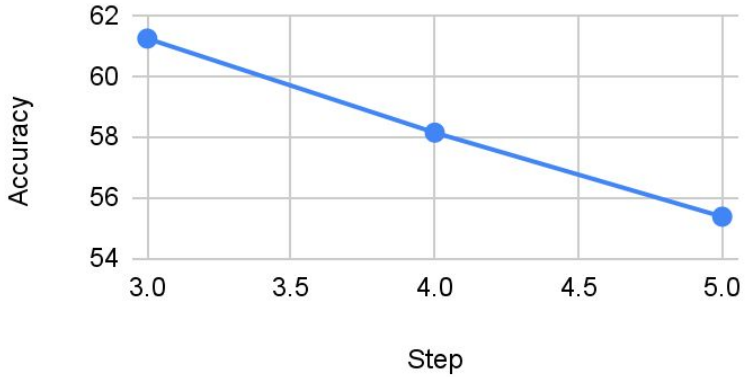
UMDFace



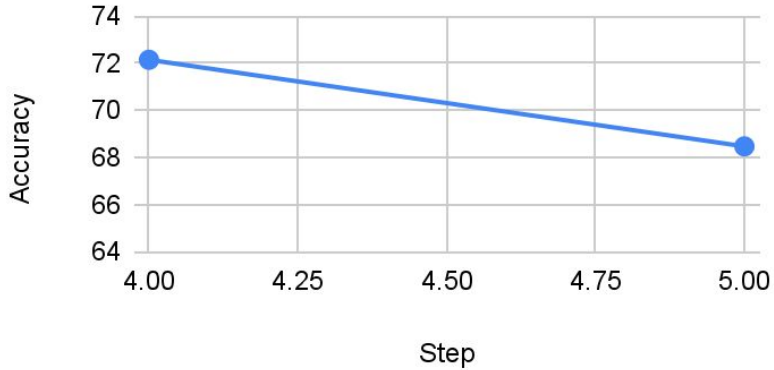
ArcFace



VGGFace



RetinaFace



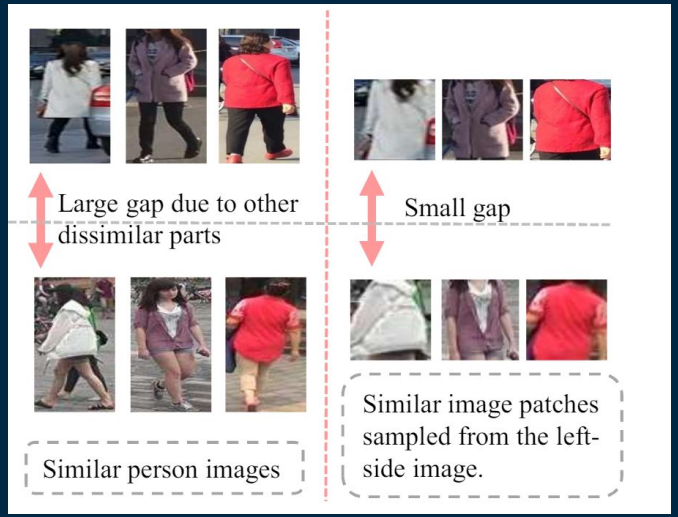
Model Modifications

PROBLEM IN THE OLD APPROACH/MOTIVATION

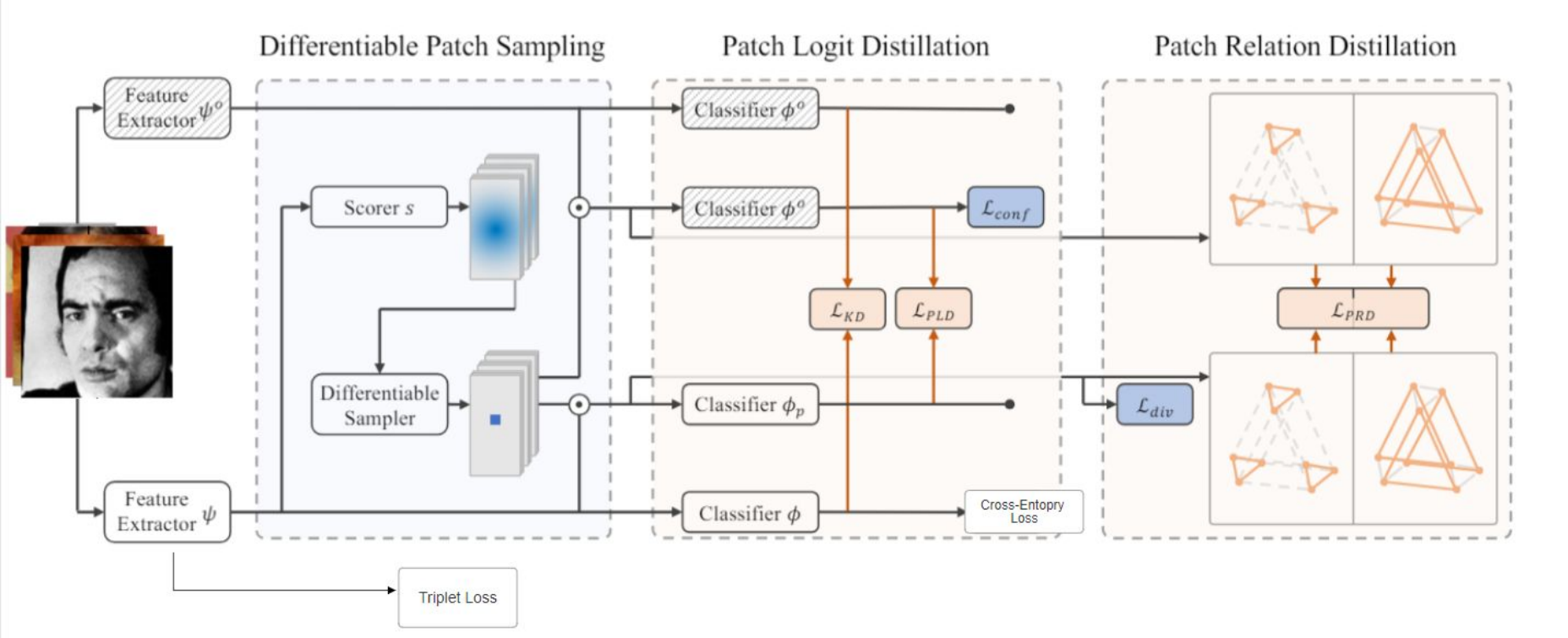
- *Our baseline model which is based on knowledge distillation relies heavily on the relatedness of the old and new tasks (LwF).*
- *Unfortunately, according to our setting, we encounter significant **distribution shifts** in the training data (as we train over different datasets).*
- *Results in a gradual error build-up to the previous datasets. This leads to a poor non-forgetting performance.*

Patch-Based Feature Learning

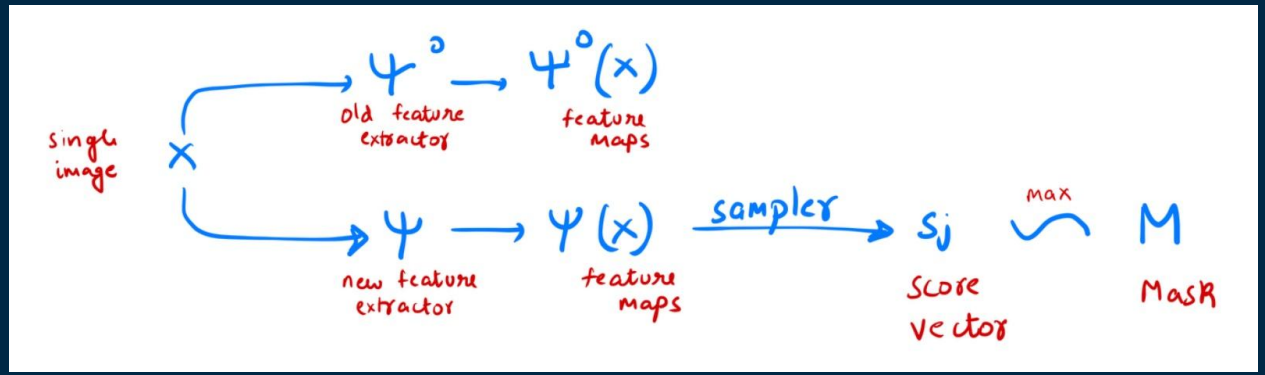
- *The distribution gap becomes much smaller when operating on patch-level features as compared to the image-level features.*
- *In order to mitigate the interference of distribution shift on knowledge distillation, we used patch-based model.*



New Model Architecture: Patch-KD (Novelty)



Patch Sampling process



$\Psi^{\circ}(x) \xrightarrow{M} p^{\circ}(x)$ old patch features

$\Psi(x) \xrightarrow{M} p(x)$ new patch features

Repeating 3 times will give you 3 patch features

$\{p_1(x), p_1^{\circ}(x)\}$ $\{p_2(x), p_2^{\circ}(x)\}$ $\{p_3(x), p_3^{\circ}(x)\}$

Confidence Loss and Diversity Loss

$$\mathcal{L}_{conf} = \frac{1}{BK} \sum_{i=1}^{B} \sum_{i=1}^{K} \mathrm{H}\left(\sigma(\phi^{o}(p_{i,r}^{o}))\right).$$

Passing the old patch features to old classifier Φ^{o} and calculating the entropy

This denotes the patch-feature's confidence on the old model

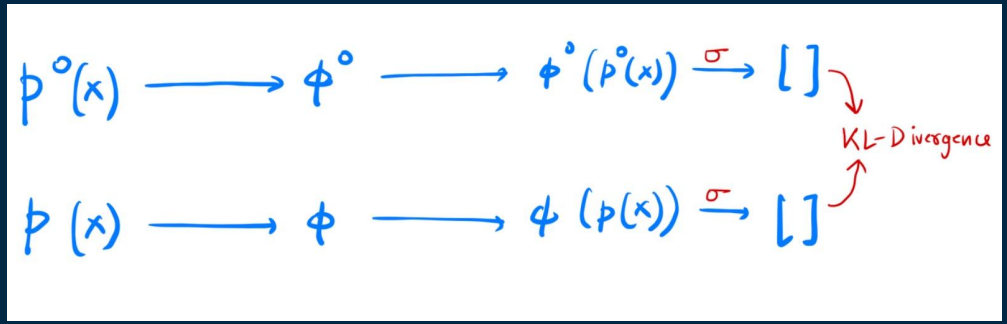
$$\mathcal{L}_{div} = \frac{1}{BK^2} \sum_{i=1}^{B} \sum_{r,s=1}^{K} \frac{\langle p_{i,r}, p_{i,s} \rangle}{\|p_{i,r}\| \|p_{i,s}\|},$$

Simply calculating the cosine similarity and penalizing the similar patches

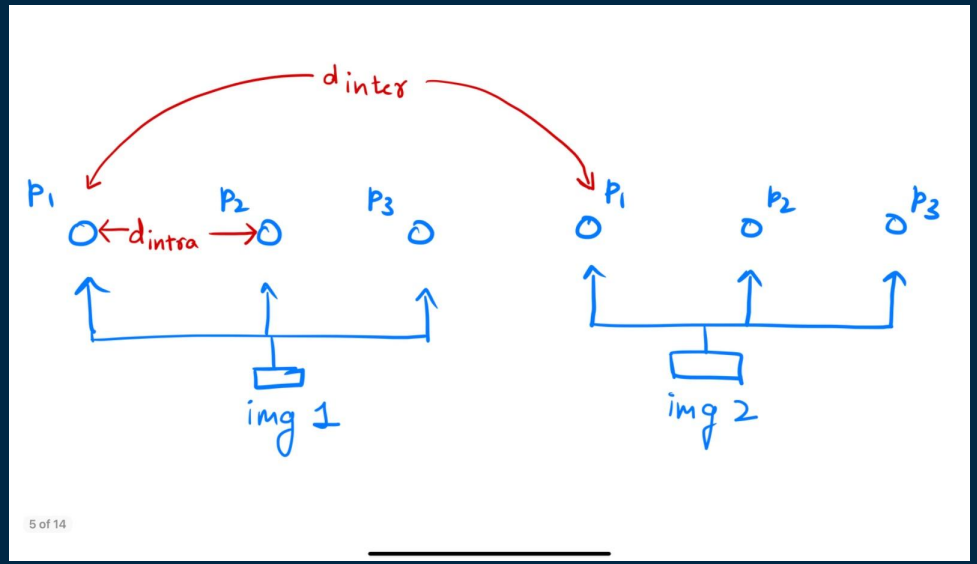
Patch Logit Distillation

$$\mathcal{L}_{PLD} = \frac{1}{BK} \sum_{i=1}^{B} \sum_{r=1}^{K} \mathrm{KL}\left(\sigma(\phi^o(p^o_{i,r})/\tau) \,\|\, \sigma(\phi_p(p_{i,r})/\tau)\right).$$

- Pass the the old and new patch features to classifiers
- Pass the raw logits to softmax to get the probabilities
- Calculate the KL-Divergence among the probabilities



Patch Relation Distillation



OBJECTIVE OF PRD

$$d^{\circ}_{intra} \overset{same}{\longleftrightarrow} d_{intra}$$

$$d^{\circ}_{inter} \overset{same}{\longleftrightarrow} d_{inter}$$

$$p^{\circ} \qquad p$$

Patch Relation Distillation

$$S_{intra} = \bigcup_{i=1}^{B} \left\{ (d(p_{i,r}, p_{i,s}), d(p^o_{i,r}, p^o_{i,s})) \;\middle|\; r, s \in [1..K], r \neq s \right\},$$

$$S_{inter} = \bigcup_{r=1}^{K} \left\{ (d(p_{i,r}, p_{j,r}), d(p^o_{i,r}, p^o_{j,r})) \;\middle|\; i, j \in [1..B], i \neq j \right\}.$$

$$\mathcal{L}_{PRD} = \frac{\sum l_\delta(d_{intra} - d^o_{intra})}{|S_{intra}|} + \frac{\sum l_\delta(d_{inter} - d^o_{inter})}{|S_{inter}|},$$

Applies a normalization because #inter-instance relations >>> #intra-instance relations

Applies huber-loss for penalizing difference between the old and new

Overall Loss function

$$\mathcal{L}_{sel} = \mathcal{L}_{conf} + \mathcal{L}_{div}.$$

$$\mathcal{L}_{base} = \mathcal{L}_{CE} + \mathcal{L}_{triplet} + \mathcal{L}_{KD}$$

$$\mathcal{L} = \mathcal{L}_{base} + \mathcal{L}_{sel} + \mathcal{L}_{PLD} + \mathcal{L}_{PRD},$$

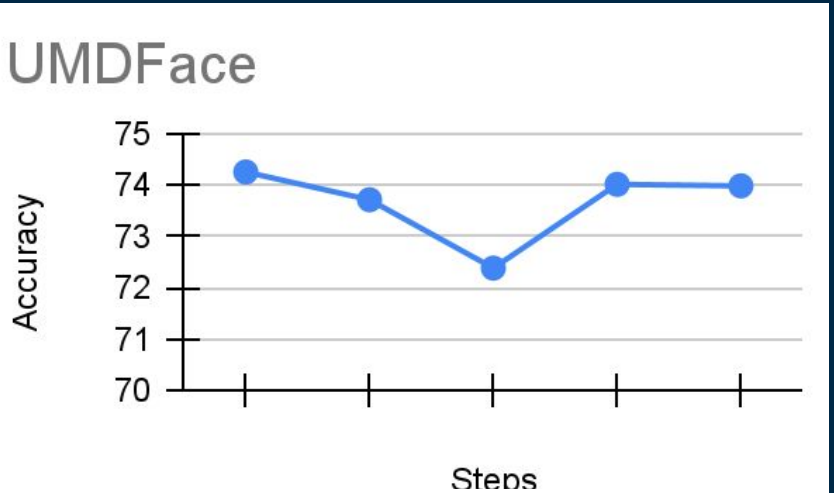
Results with all loss functions on (for the Patch-based Model)

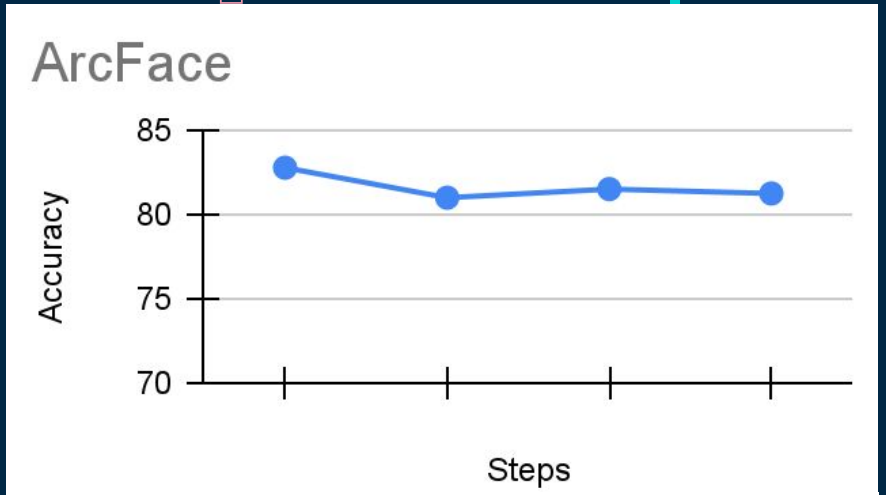
Test →

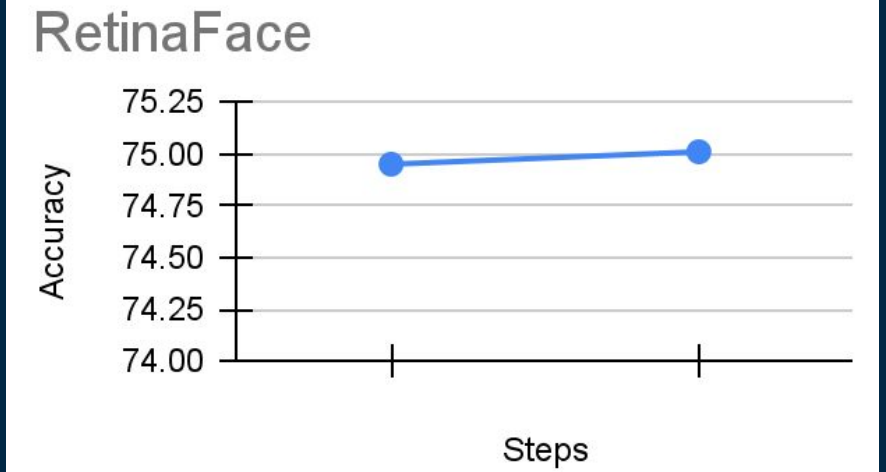
Train ↓

	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**74.26**				
ArcFace	**73.72**	**82.81**			
VGGFace	**72.39**	**81.03**	**65.16**		
RetinaFace	**74.02**	**81.54**	**63.09**	**74.95**	
CasiaFace	**73.99**	**81.28**	**62.01**	**75.01**	**55.8**
	-0.27	-1.53	-3.15	-0.06	

Results with all loss functions on







Phase 3

Ablation studies on loss functions

Removing the Loss functions for Differentiable Patch Sampler

Test →

Train ↓

-conf -div	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**73.57**				
ArcFace	**72**	**80.42**			
VGGFace	**71.68**	**79.81**	**64.22**		
RetinaFace	**73.08**	**80.56**	**62.58**	**74.63**	
CasiaFace	**72.92**	**80.08**	**61.56**	**74.2**	**55.3**
	-0.65	-0.34	-2.66	-0.43	

Ablation studies on loss functions

Removing the Loss function for Patch Logit Distillation

Test →

Train ↓

-pld_loss	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**74.29**				
ArcFace	**73.78**	**82.91**			
VGGFace	**73.17**	**81.07**	**65.36**		
RetinaFace	**74.09**	**81.98**	**64.07**	**74.92**	
CasiaFace	**74.02**	**81.72**	**62.2**	**75.07**	**55.89**
	-0.27	-1.19	-3.16	-0.15	

Ablation studies on loss functions

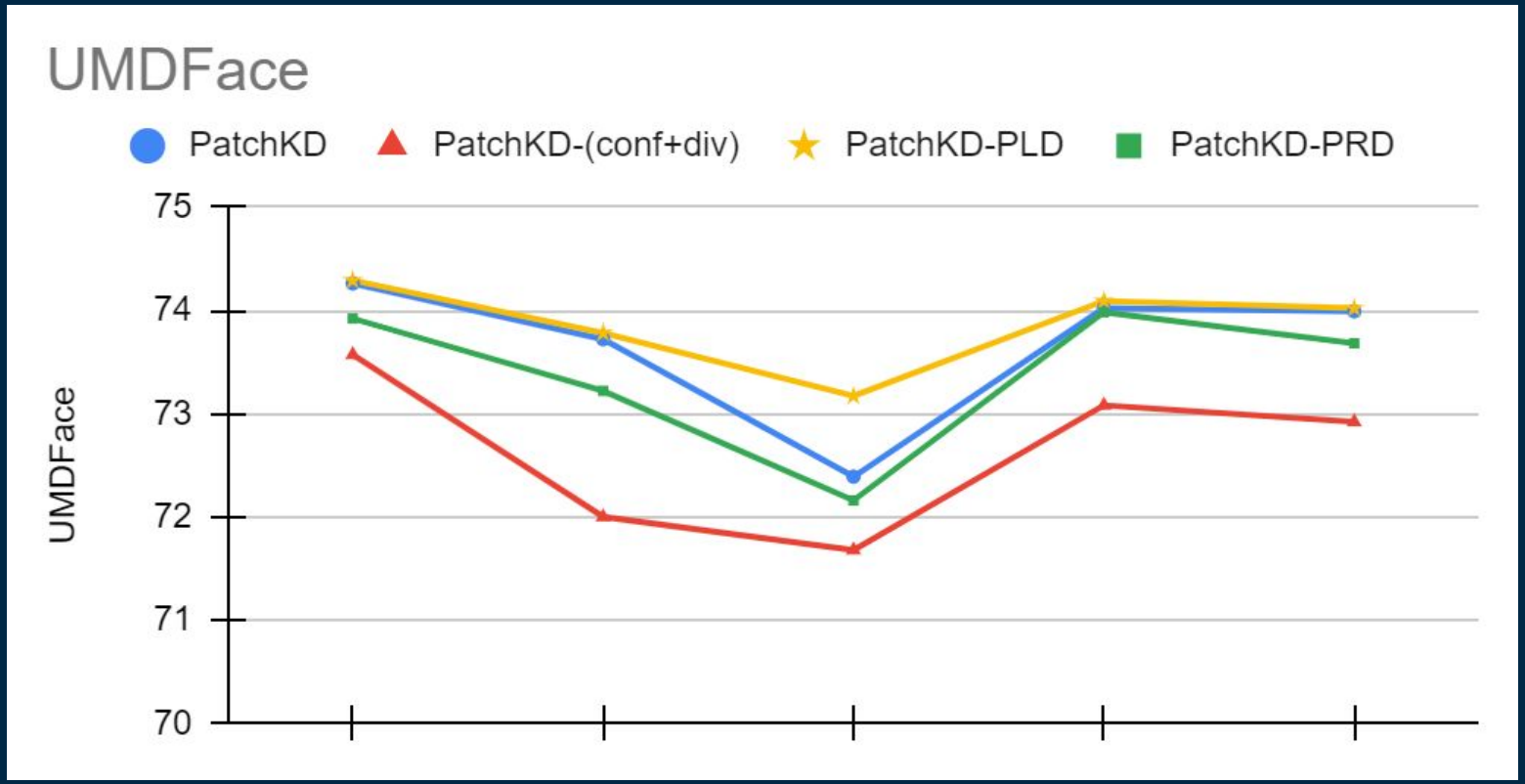
Removing the Loss function for Patch Relation Distillation

Test →

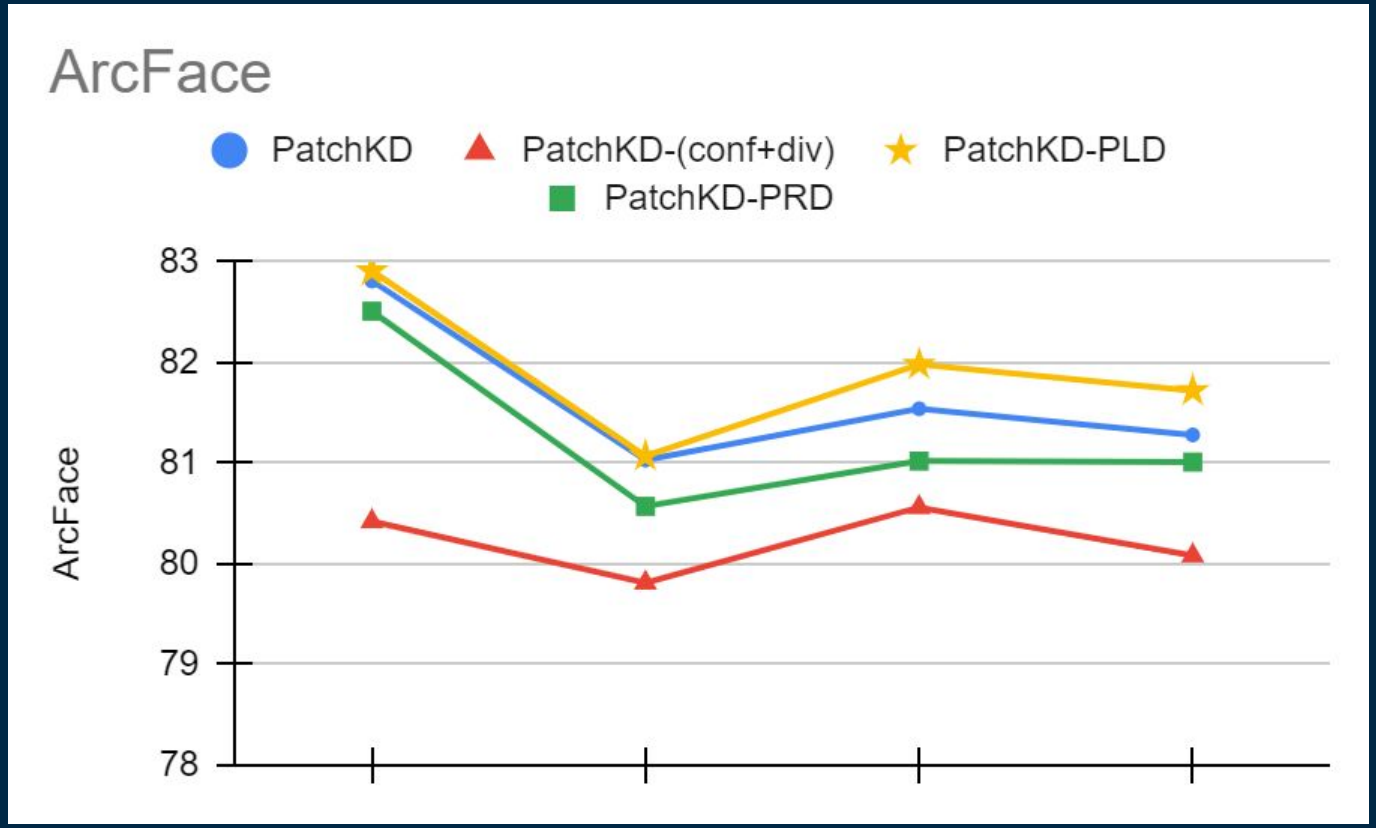
Train ↓

-prd_loss	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**73.92**				
ArcFace	**73.22**	**82.51**			
VGGFace	**72.16**	**80.57**	**65**		
RetinaFace	**73.98**	**81.02**	**62.85**	**73.82**	
CasiaFace	**73.68**	**81.01**	**61.77**	**74.63**	**55.13**
	-0.24	-1.5	-3.23	-0.81	

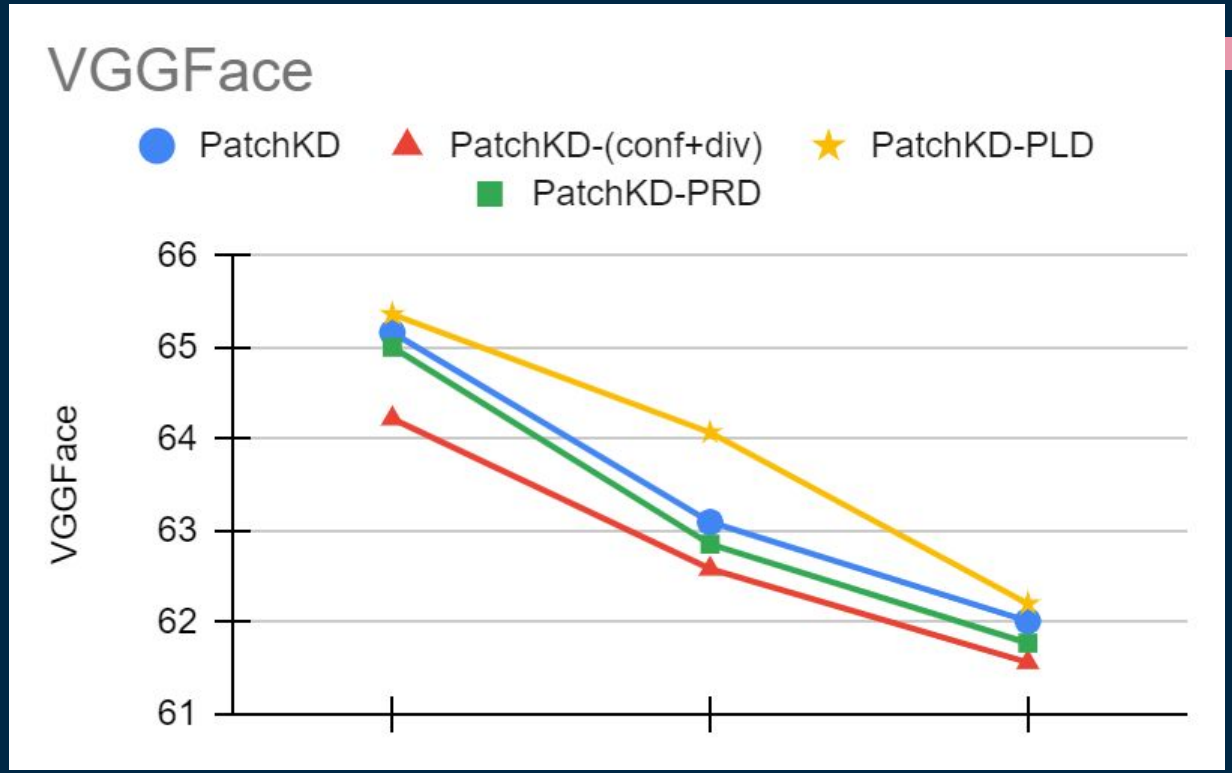
Results with all loss functions on



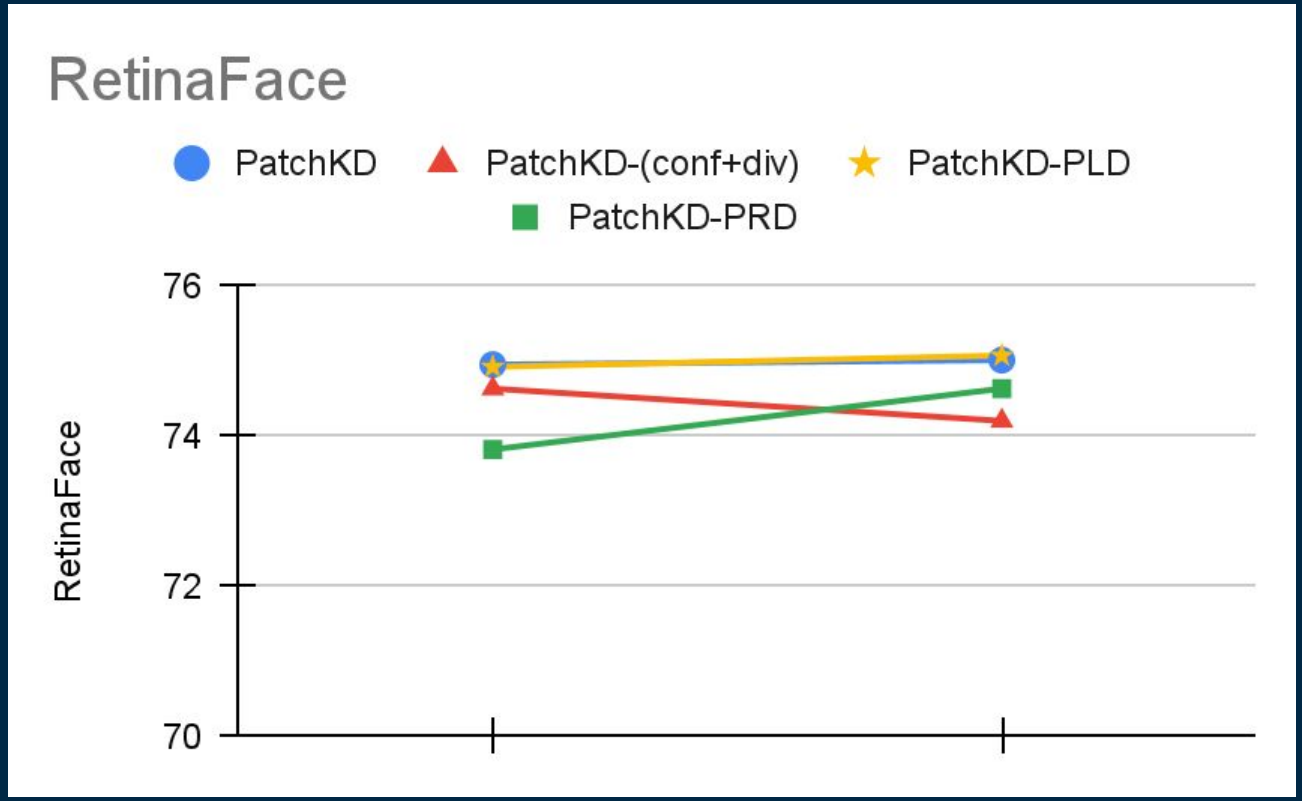
Results with all loss functions on



Results with all loss functions on



Results with all loss functions on



What Happened?

Visualization (Patch Logit Distillation)

Example:

Model 3 patch visualization on a class

Model 4 patch visualization on a class

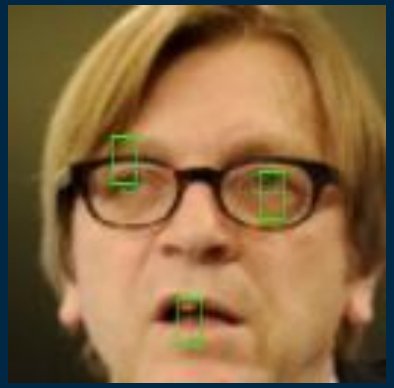
In image one, the previous model learns patches that’re irrelevant for distillation, and then using PLD the new model is enforced to learn the same patches.

This accumulates the error as model is trained on new classes.

$$\mathcal{L}_{PLD} = \frac{1}{BK} \sum_{i=1}^{B} \sum_{r=1}^{K} \mathrm{KL}\left(\sigma(\phi^{o}(p^{o}_{i,r})/\tau) \,\|\, \sigma(\phi_{p}(p_{i,r})/\tau)\right).$$

Visualization (Patch Relation Distillation)

Example:



Model 6 patch visualization on a class

Model 7 patch visualization on a class

In image one, the model learned the patches near eyes and mouth, and PRD enforced the new model to keep the distance between the new patches same as before.

$$\mathcal{L}_{PRD} = \frac{\sum l_\delta(d_{intra} - d^o_{intra})}{|S_{intra}|} + \frac{\sum l_\delta(d_{inter} - d^o_{inter})}{|S_{inter}|},$$

Visualization (Confidence Loss)

Example:

Model 2 batch 40 patch visualization on a class

Model 2 batch 41 patch visualization on a class

In image one, the model learned the patches near eyes and mouth, and PRD enforced the new model to keep the distance between the new patches same as before.

Visualization (Diversity Loss)

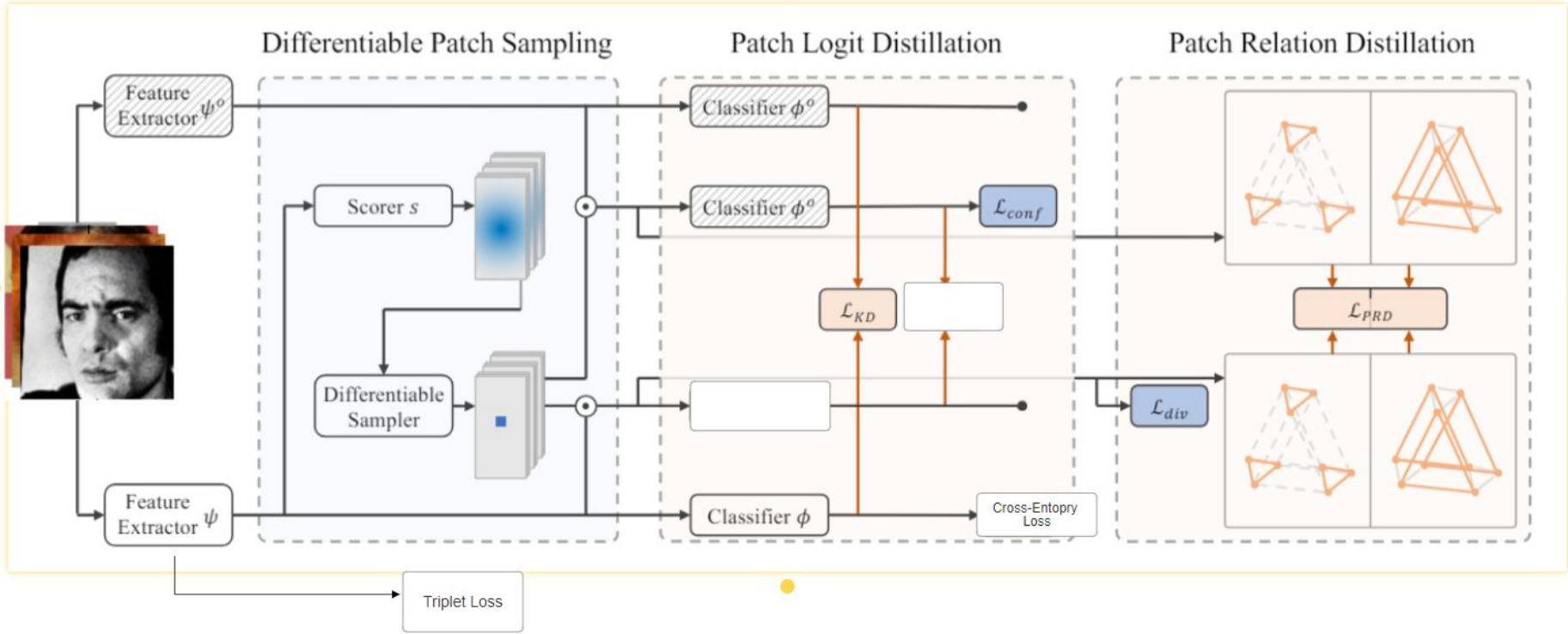
Example:

Training without div loss,
Model 12

Training with div loss,
Model 12

Div loss enforce the model to learned dissimilar patches, patches that have different features encoded.

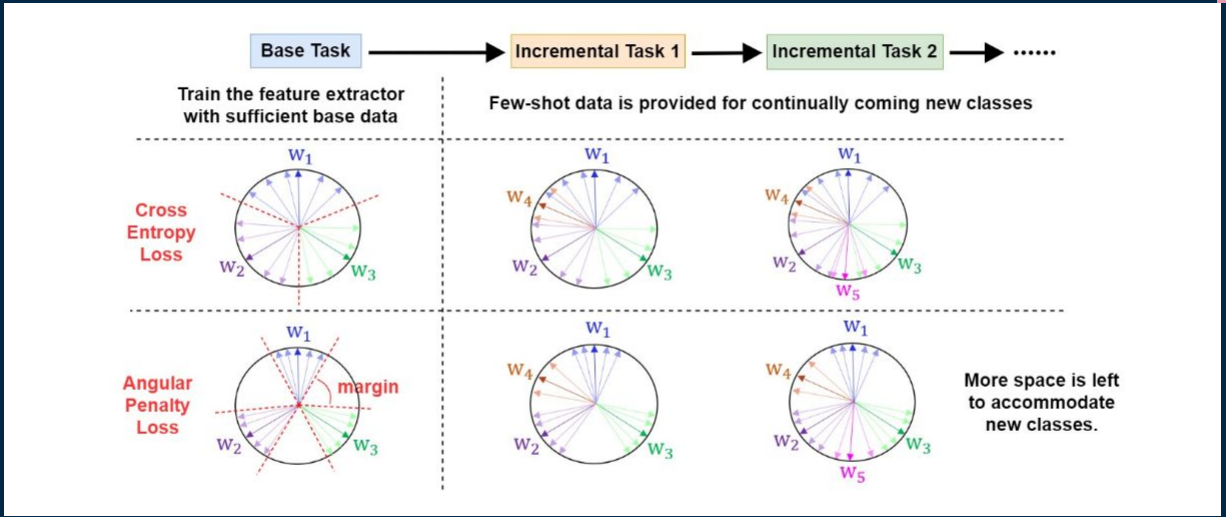
Updated Model Architecture



$$\mathcal{L} = \mathcal{L}_{base} + \mathcal{L}_{sel} + \mathcal{L}_{PRD},$$

Phase 4

Adding Few-Shots specific loss function (Novelty)



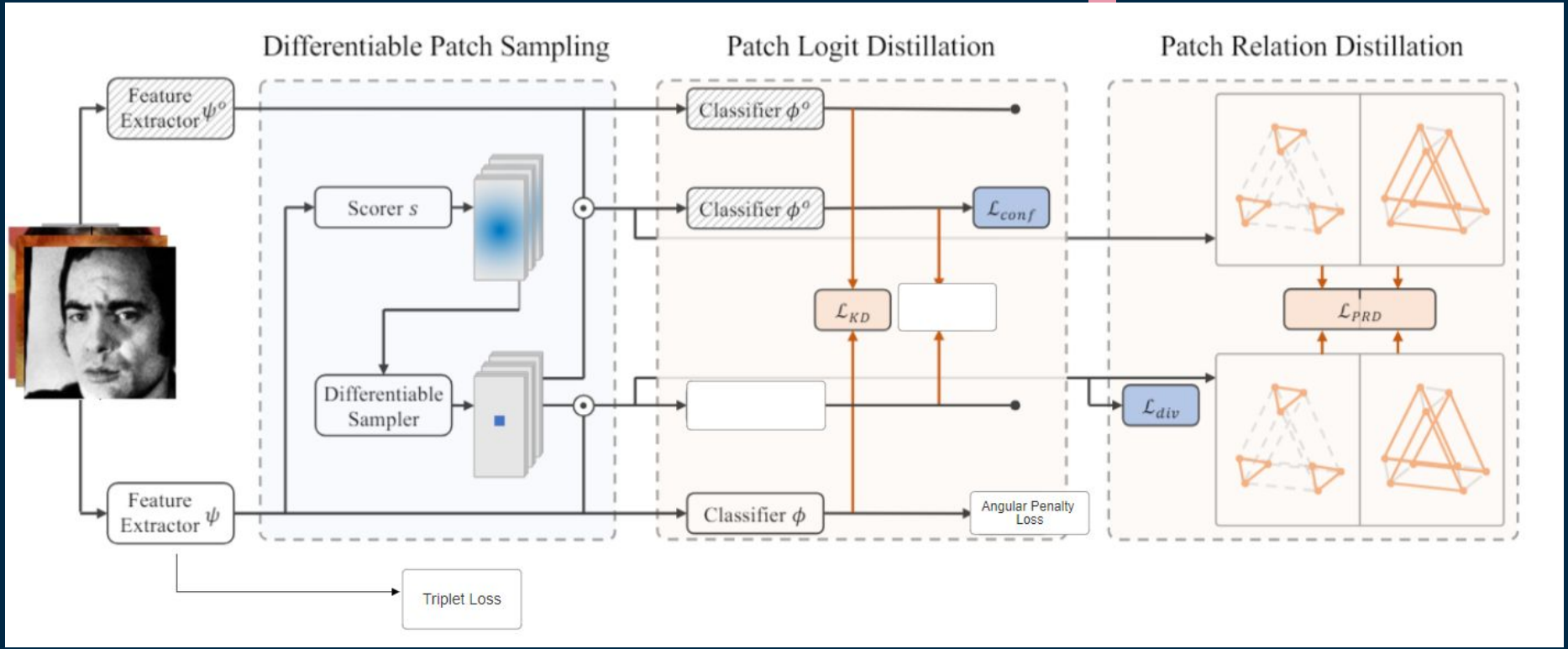
FSCIL from an open-set perspective using Angular Penalty Loss

$$L_{AP} = -\frac{1}{N}\sum_{j=1}^{N}\log\left(\frac{e^{s(\cos(\theta_j)-m)}}{e^{s(\cos(\theta_j)-m)} + \sum_{i\neq j} e^{s\cos(\theta_i)}}\right)$$

s - scale factor = 30

m - cosine margin = 0.4

Final Model Architecture



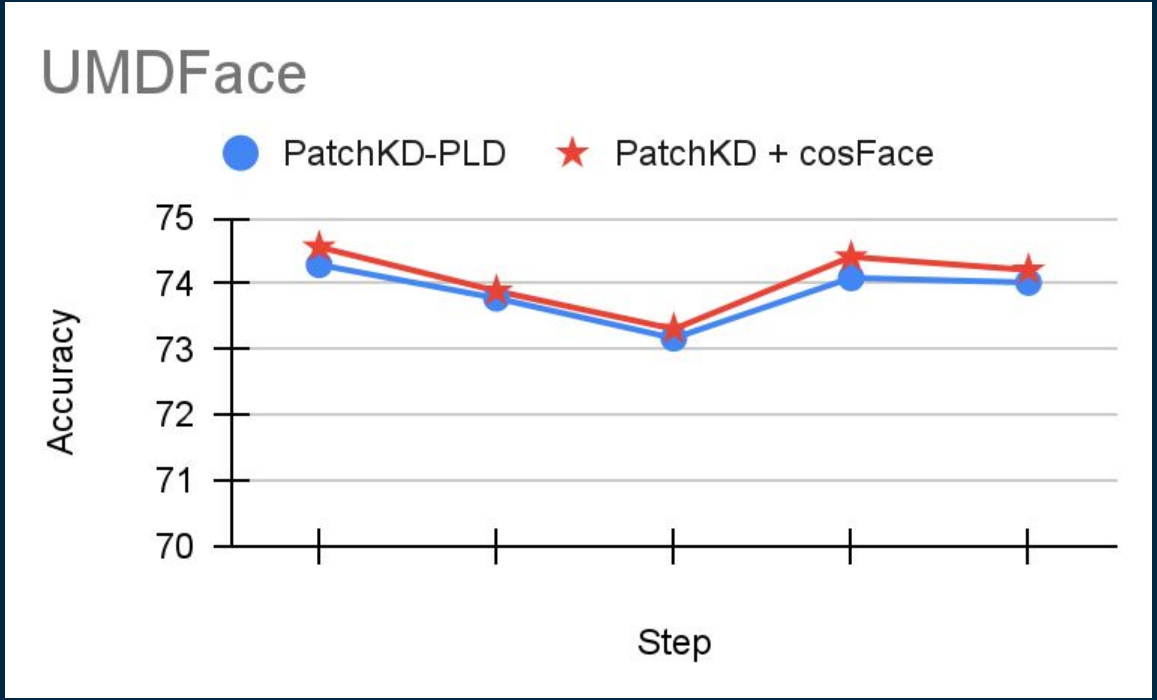
Results at T=5, K=4 (for the final Model with cosFace)

Test →

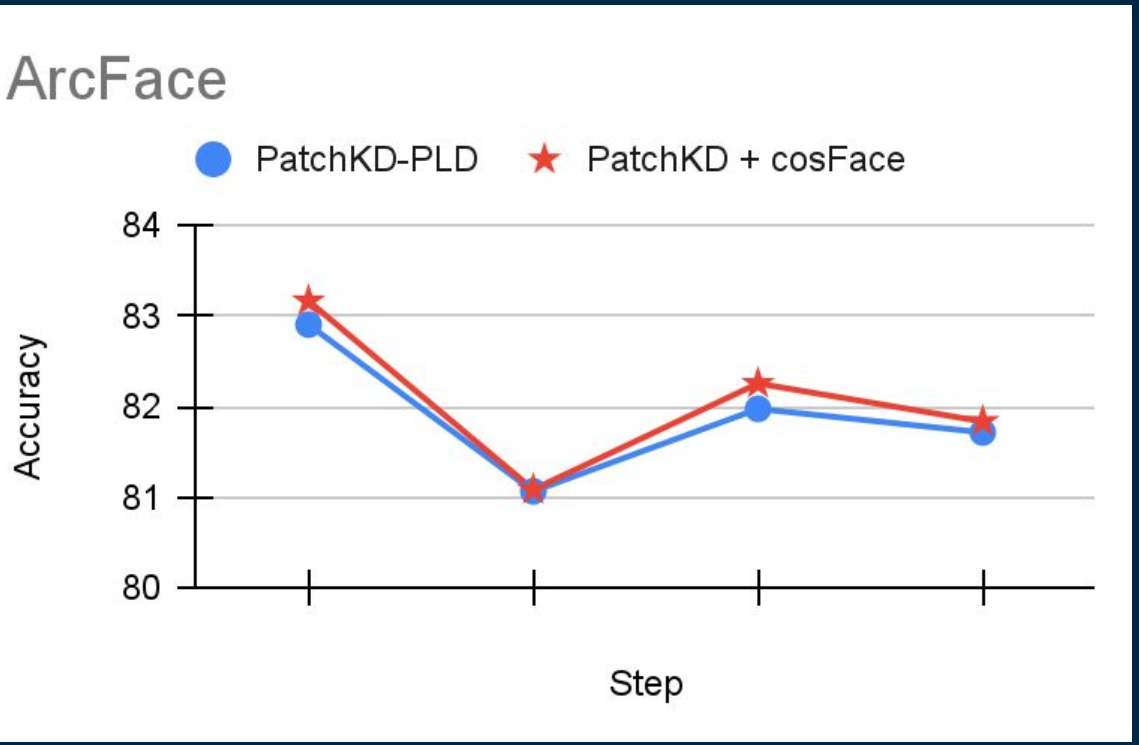
Train ↓

cosface	UMDFace	ArcFace	VGGFace	RetinaFace	CasiaFace
UMDFace	**74.56**				
ArcFace	**73.89**	**83.17**			
VGGFace	**73.31**	**81.09**	**65.91**		
RetinaFace	**74.41**	**82.26**	**64.96**	**76.01**	
CasiaFace	**74.21**	**81.84**	**62.71**	**75.52**	**57.02**
	-0.35	-0.67	-3.2	-0.49	

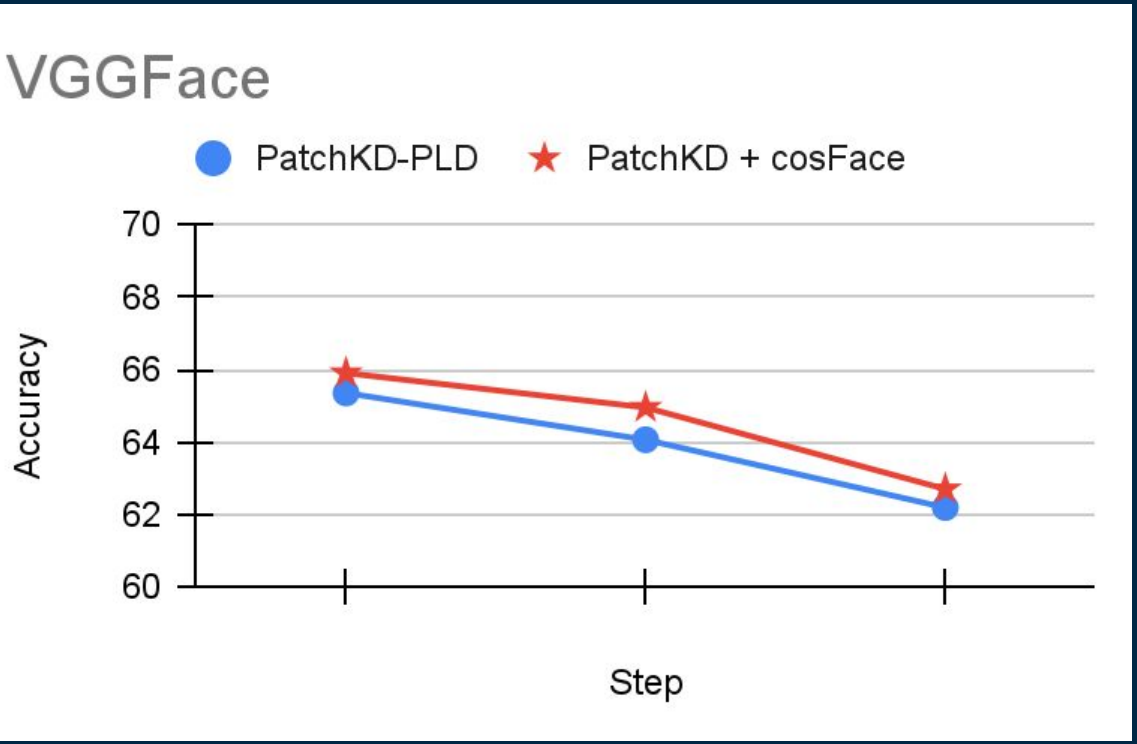
Results


UMDFace
PatchKD-PLD
PatchKD + cosFace
75
74
73
72
71
70
Accuracy
Step

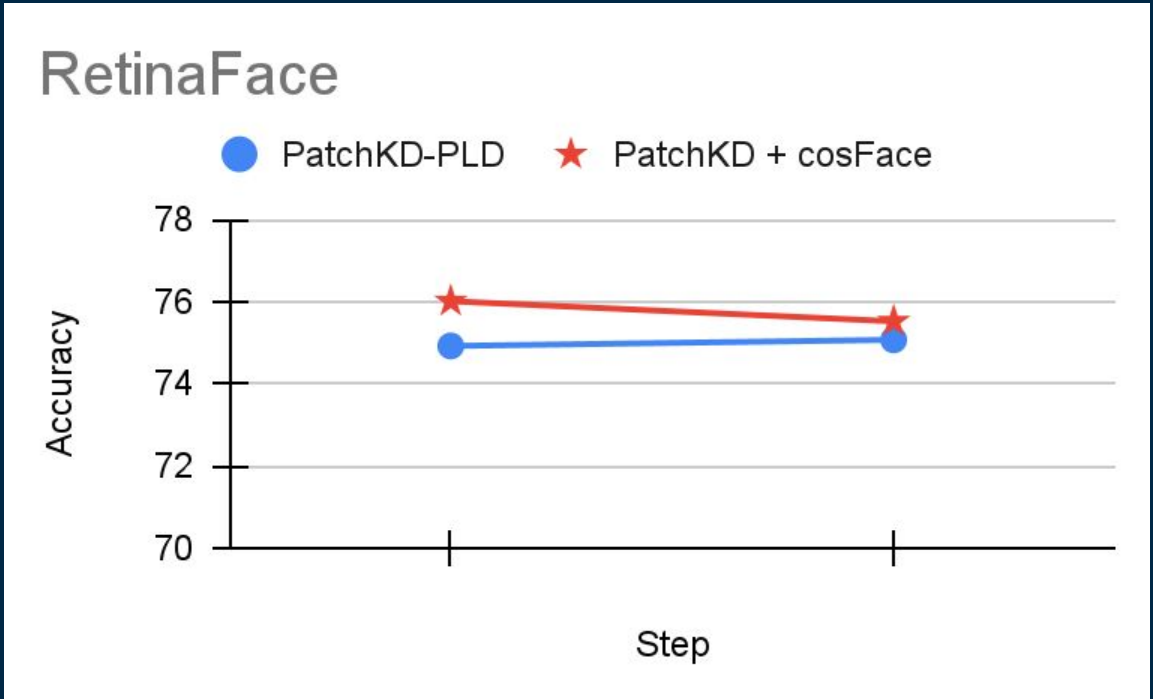
Results


ArcFace
PatchKD-PLD
PatchKD + cosFace
84
83
82
81
80
Accuracy
Step

Results

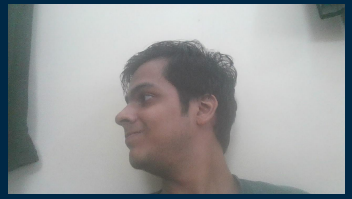
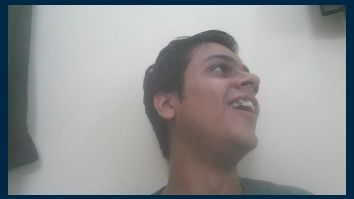

VGGFace
PatchKD-PLD
PatchKD + cosFace
70
68
66
64
62
60
Accuracy
Step

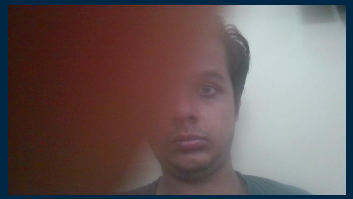
Results

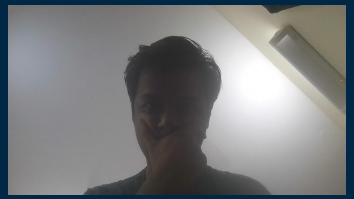
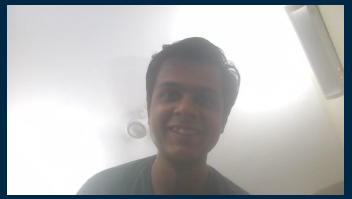

RetinaFace
PatchKD-PLD
PatchKD + cosFace
78
76
74
72
70
Accuracy
Step

Demo

Demo



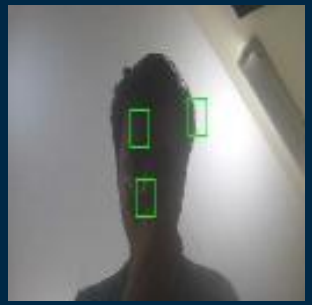
Used these 8 images out of which the model randomly picks K=4 images to train, one image in the query set and rest of the images in the gallery set to match with.

Results of Demo

- Setting was T=5, K=4, Q=5.
- Added the image to one of the classes in the dataset

Trained in step 1-5 (D1)	Experimented 20 times	16/20 detections
Trained in step 10-15 (D3)	Experimented 5 times	2/5 detections
Trained in step 20-25 (D5)	Experimented 5 times	1/5 detections

Some of the Patch Visualizations



Future Work

- Modifying the PLD loss function such that it can distill knowledge in a more robust way.
- Using a feature based distillation rather than a logit-based distillation to learn more robust representations and make our model scalable.
- Doing testing at T=10, T=20 steps and K=2, K=1 shots to check the implications of cosFace loss function and get more theoretical insights on the patch-based loss functions.

We're aiming to submit our work in ICPR with deadline on June 30.

Thank You

References

- https://avalanche.continualai.org
- Measuring Catastrophic Forgetting in Neural Networks
- Spot® - The Agile Mobile Robot | Boston Dynamics
- Deep Class-Incremental-Learning: A Survey