UNIVERSITÄT MANNHEIM

Practical aspects and implications of certified robustness

Shashank Agnihotri and Prof. Dr.-Ing Margret Keuper

Introduction

- **Prof. Dr.-Ing. Margret Keuper**
- **Professor for Machine Learning**
- Research Interests:
  - Machine Learning and Computer Vision
  - Video Segmentation
  - Motion Analysis
- Room: B1.18
- email: keuper@uni-mannheim.de

Introduction

- **Shashank Agnihotri**
- PhD student
- Research Interests:
  - Robustness of Deep Neural Network Architectures
  - Network Pruning
  - Neural Architecture Search
- Room: C0.02
- email: shashank.agnihotri@uni-mannheim.de

What is Certified Robustness?

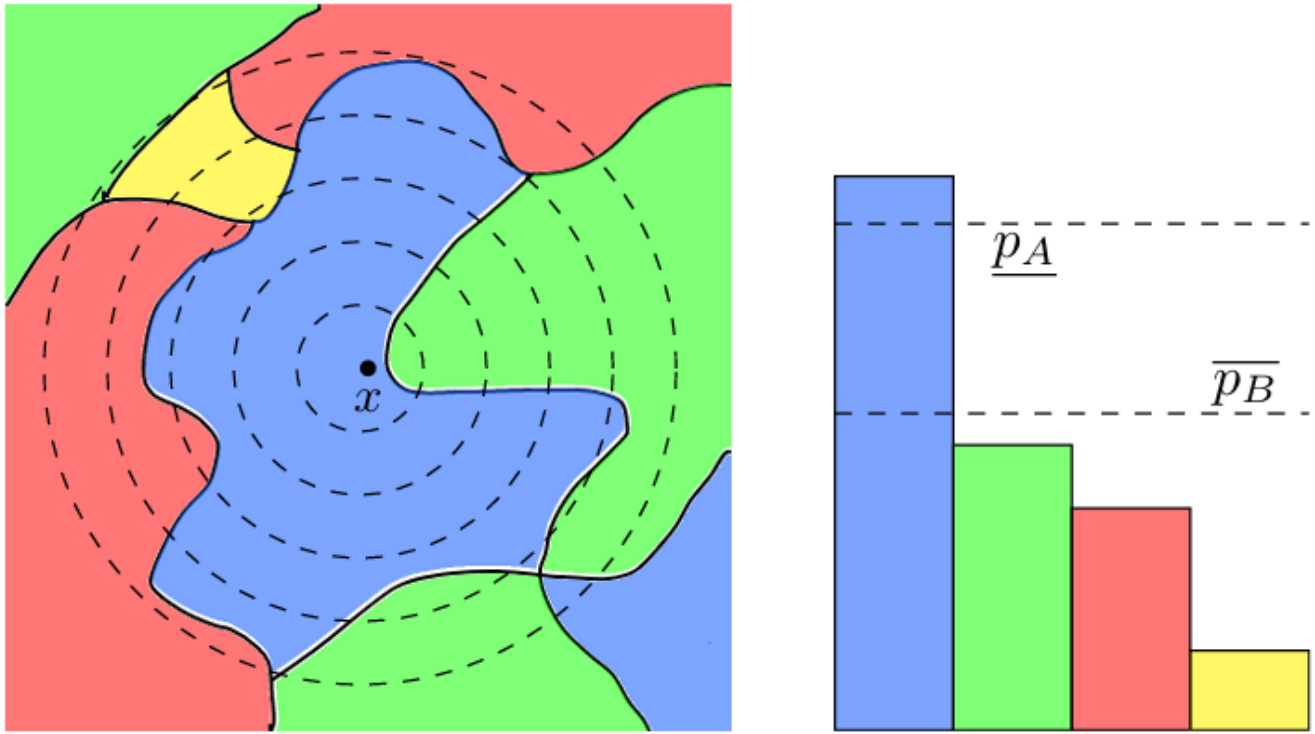
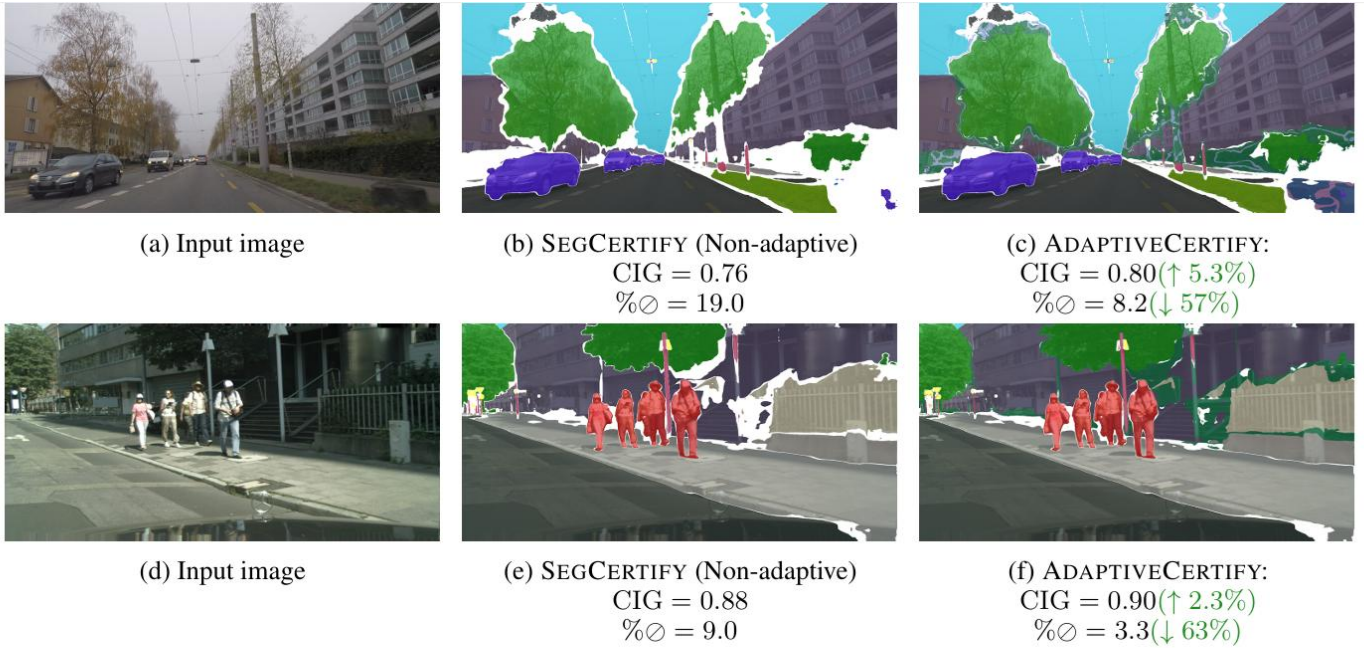


Figure source: Cohen, Jeremy, Elan Rosenfeld, and Zico Kolter. "Certified adversarial robustness via randomized smoothing." *international conference on machine learning*. PMLR, 2019.

How is certified robustness useful?



(a) Input image

(b) SEGCERTIFY (Non-adaptive)
CIG $= 0.76$
$\%\oslash = 19.0$

(c) ADAPTIVECERTIFY:
CIG $= 0.80(\uparrow 5.3\%)$
$\%\oslash = 8.2(\downarrow 57\%)$

(d) Input image

(e) SEGCERTIFY (Non-adaptive)
CIG $= 0.88$
$\%\oslash = 9.0$

(f) ADAPTIVECERTIFY:
CIG $= 0.90(\uparrow 2.3\%)$
$\%\oslash = 3.3(\downarrow 63\%)$

Figure 1: The certified segmentation outputs on input images (a) and (d) from SEGCERTIFY in (b) and (e), and ADAPTIVECERTIFY in (c) and (f) with their corresponding Certified Information Gain (CIG) and abstain rate $\%\oslash$. Our method provides more meaningful certified output in pixels the state-of-the-art abstains from (white pixels), with a much lower abstain rate, and higher CIG. The segmentation color palette can be found in Figure 2.

Figure source: Anani, Alaa, et al. "Adaptive Hierarchical Certification for Segmentation using Randomized Smoothing." *Forty-first International Conference on Machine Learning*.

However, their practical applicability is not studied!

What will we do?

Project Tasks

- Adversarial Attacks: Patch Attacks and Image-wide attacks:
  - PGD, APGD, AutoAttack and CosPGD (for semantic segmentation)
- OOD Robustness studies:
  - using 2D and 3D Common Corruptions,
  - in case of semantic segmentation: inference on Zurich night and ACDC.

Important Certified Robustness Works we will use

- Xiang, Chong, Saeed Mahloujifar, and Prateek Mittal. "{PatchCleanser}: Certifiably robust defense against adversarial patches for any image classifier." *31st USENIX Security Symposium (USENIX Security 22)*. 2022.
- Anani, Alaa, et al. "Adaptive Hierarchical Certification for Segmentation using Randomized Smoothing." *Forty-first International Conference on Machine Learning*
- Cohen, Jeremy, Elan Rosenfeld, and Zico Kolter. "Certified adversarial robustness via randomized smoothing." *international conference on machine learning*. PMLR, 2019.
- C. -C. Kao, C. -S. Lu and C. -M. Yu, "Image Forensics Strikes Back: Defense Against Adversarial Patch," 2024 IEEE International Conference on Visual Communications and Image Processing (VCIP), Tokyo, Japan, 2024, pp. 1-5, doi: 10.1109/VCIP63160.2024.10849849.
- Xiang, Chong, et al. "Certifiably Robust RAG against Retrieval Corruption." arXiv preprint arXiv:2405.15556 (2024).
- PatchDEMUX: A Certifiably Robust Framework for Multi-label Classifiers Against Adversarial Patches, Dennis Jacob, Chong Xiang, Prateek Mittal
- Xiang, Chong, and Prateek Mittal. "PATCHGUARD++: EFFICIENT PROVABLE ATTACK DE-TECTION AGAINST ADVERSARIAL PATCHES."
- Xiang, Chong, et al. "{PatchGuard}: A provably robust defense against adversarial patches via small receptive fields and masking." 30th USENIX Security Symposium (USENIX Security 21). 2021.

Important details

- Language: English
- Duration: 1 semester
- Min/max number of participants: 4-6
- Prerequisites: Intermediate Python and Pytorch skills
- Applicable to MMDS: yes
- Online: possible (preferred)

Further Questions

Please contact:
Shashank Agnihotri

Google chat:
shashanksagnihotri@gmail.com

Email:
shashank.agnihotri@uni-mannheim.de

Message header:
“Team Project Fall 2025”

Please also include:
Proficiency in Python and Pytorch


Thank
You!

APPENDIX

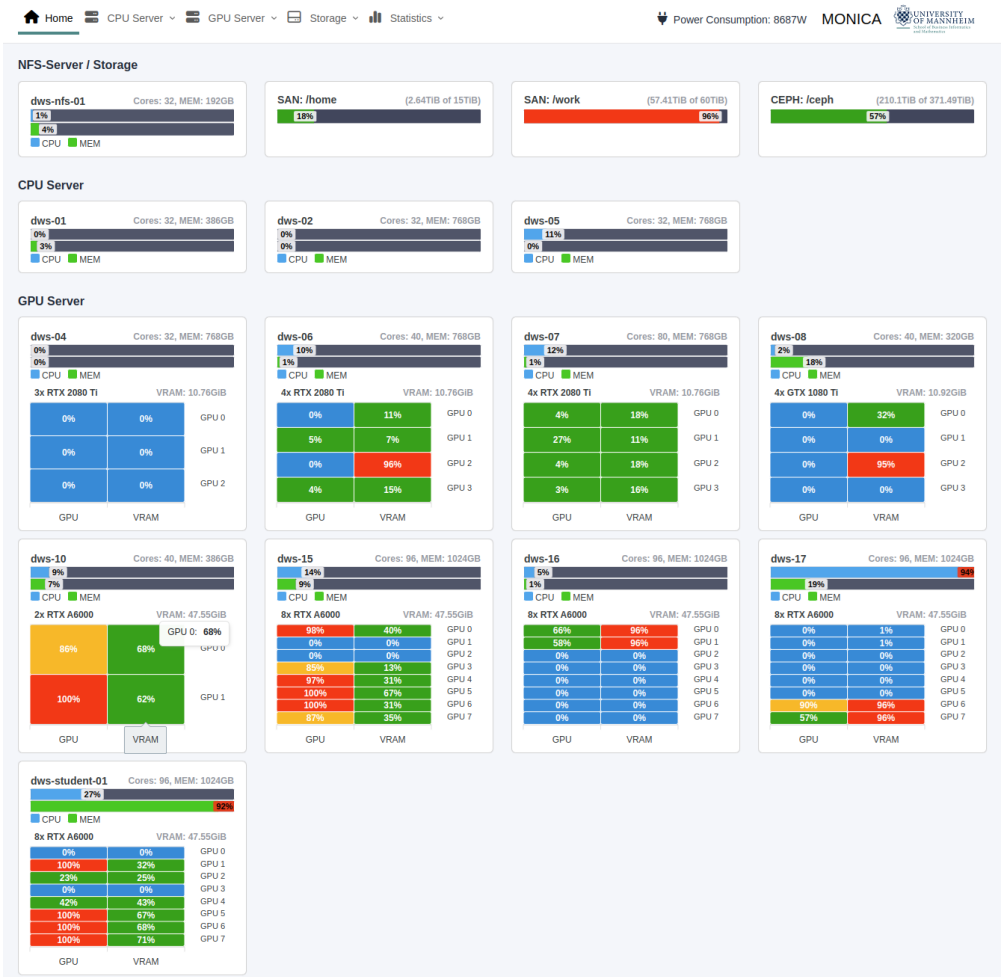
Meeting Schedule

- Weekly Meetings: Time and Day to be decided

Coding Ethics

- Maintain a Project github
- Work on branches
- Please push changes to your branch after EVERY SITTING

What is a cluster?



How to access the cluster?

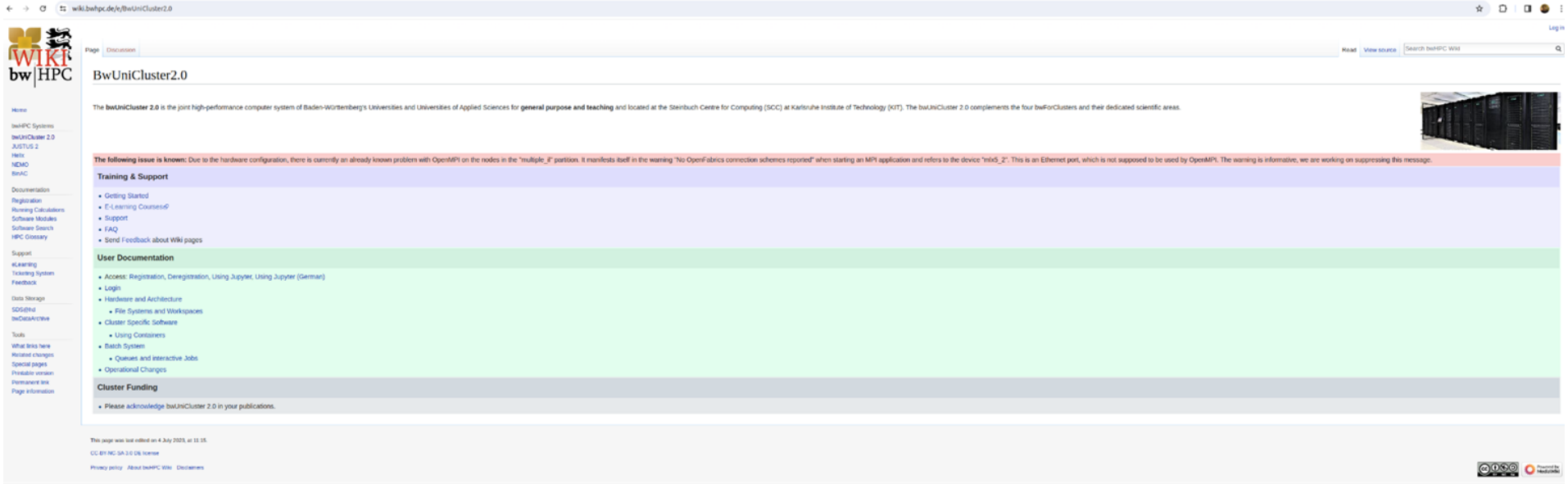
Be inside the university network

Or

Use Uni Mannheim VPN

What do we use?

https://wiki.bwhpc.de/e/BwUniCluster2.0



How do we login?

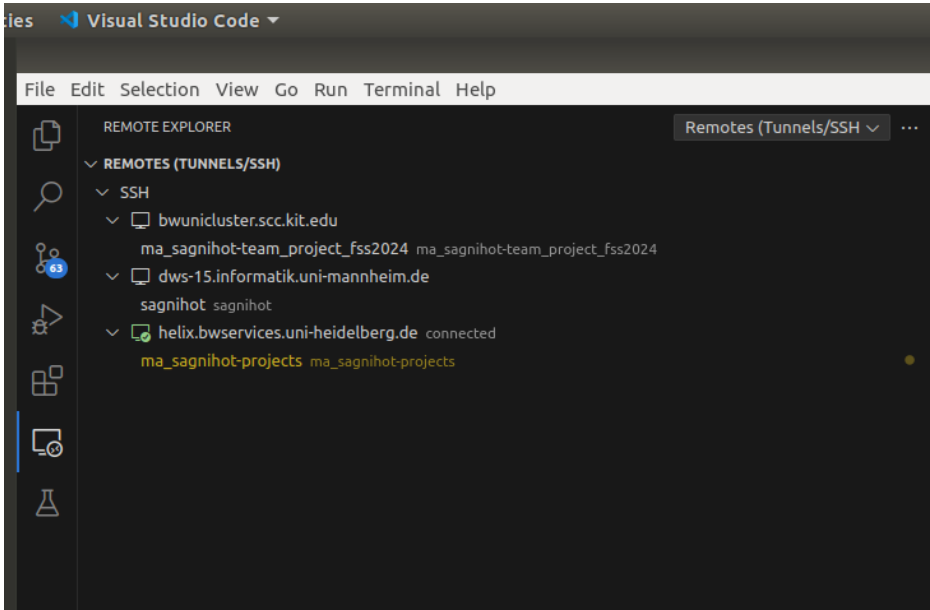
https://wiki.bwhpc.de/e/BwUniCluster2.0/Login

ssh ma_<user_name>@bwunicluster.scc.kit.edu

Example: my username is sagnihot

So I use:

ssh ma_sagnihot@bwunicluster.scc.kit.edu



Where do we store our files and data?

Please use workspaces.

ws_allocate team_project_fss2024 60 -m <email_id> -r 4
And then cd to the path

```
(base) [ma_sagnihot@uc2n995 ~]$ ws_allocate team_project_fss2024 60 -m shashank.agnihotri@uni-mannheim.de -r 3
Info: creating workspace.
/pfs/work7/workspace/scratch/ma_sagnihot-team_project_fss2024
remaining extensions  : 3
remaining time in days: 60
(base) [ma_sagnihot@uc2n995 ~]$ cd /pfs/work7/workspace/scratch/ma_sagnihot-team_project_fss2024
(base) [ma_sagnihot@uc2n995 ma_sagnihot-team_project_fss2024]$ ws_list
id: projects
     workspace directory  : /pfs/work7/workspace/scratch/ma_sagnihot-projects
     remaining time       : 58 days 4 hours
     creation time        : Tue Feb 27 16:27:15 2024
     expiration date      : Sat Apr 27 17:27:15 2024
     filesystem name      : pfs5wor7
     available extensions : 3
id: team_project_fss2024
     workspace directory  : /pfs/work7/workspace/scratch/ma_sagnihot-team_project_fss2024
     remaining time       : 59 days 23 hours
     creation time        : Thu Feb 29 12:03:38 2024
     expiration date      : Mon Apr 29 13:03:38 2024
     filesystem name      : pfs5wor7
     available extensions : 3
(base) [ma_sagnihot@uc2n995 ma_sagnihot-team_project_fss2024]$
```

How to setup anaconda?

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh

chmod +x Miniconda3-latest-Linux-x86_64.sh

./Miniconda3-latest-Linux-x86_64.sh

Yes

<path to your workspace>/miniconda3

Example: /pfs/work7/workspace/scratch/ma_sagnihot-team_project_fss2024/miniconda3

When done: source ~/.bashrc

What is SLURM?

https://wiki.bwhpc.de/e/BwUniCluster2.0/Slurm

1.2 Slurm Commands (excerpt)

Some of the most used Slurm commands for non-administrators working on bwUniCluster 2.0.

Slurm commands	Brief explanation
sbatch	Submits a job and queues it in an input queue [sbatch]
scontrol show job	Displays detailed job state information [scontrol]
squeue	Displays information about active, eligible, blocked, and/or recently completed jobs [squeue]
squeue --start	Returns start time of submitted job or requested resources [squeue]
sinfo_t_idle	Shows what resources are available for immediate use [sinfo]
scancel	Cancels a job (obsoleted!) [scancel]

- Slurm Tutorials
- Slurm command/option summary (2 pages)
- Slurm Commands

What do we use on this cluster?

bwUniCluster 2.0 sbatch -p *queue*				
queue	**node**	**default resources**	**minimum resources**	**maximum resources**
dev_gpu_4_a100	IceLake + A100	time=10, mem-per-gpu=127500mb, cpus-per-gpu=16		time=30, nodes=1, mem=510000mb, ntasks-per-node=64, (threads-per-core=2)
gpu_4_a100	IceLake + A100	time=10, mem-per-gpu=127500mb, cpus-per-gpu=16		time=48:00:00, nodes=9, mem=510000mb, ntasks-per-node=64, (threads-per-core=2)
gpu_4_h100	IceLake + H100	time=10, mem-per-gpu=127500mb, cpus-per-gpu=16		time=48:00:00, nodes=5, mem=510000mb, ntasks-per-node=64, (threads-per-core=2)
fat		time=10, mem-per-cpu=18750mb	mem=180001mb	time=72:00:00, nodes=1, mem=3000000mb, ntasks-per-node=96, (threads-per-core=2)
dev_gpu_4	gpu4	time=10, mem-per-gpu=94000mb, cpus-per-gpu=10		time=30, nodes=1, mem=376000, ntasks-per-node=40, (threads-per-core=2) 1 node is reserved for this queue Only for development, i.e. debugging or performance optimization ...
gpu_4	gpu4	time=10, mem-per-gpu=94000mb, cpus-per-gpu=10		time=48:00:00, mem=376000, nodes=14, ntasks-per-node=40, (threads-per-core=2)
gpu_8	gpu8	time=10, mem-per-cpu=94000mb, cpus-per-gpu=10		time=48:00:00, mem=752000, nodes=10, ntasks-per-node=40, (threads-per-core=2)

Get a gpu for debugging?

For 32GB Nvidia Tesla V100:

```
srun -p dev_gpu_4 --gres=gpu:1 --nodes=1 --ntasks-per-node=1 --time=0:30:00 --mem=55gb --cpus-per-task=16 --pty bash
```

Or

For 80GB Nvidia A100:

```
srun -p dev_gpu_4_a100 --gres=gpu:1 --nodes=1 --ntasks-per-node=1 --time=0:30:00 --mem=55gb --cpus-per-task=16 --pty bash
```

How to submit a job?

sbatch <path_to_bash_script>

```
(base) [ma_sagnihot@o05i14 freq-restormer]$ sbatch scripts/train_flc_pgd.sh
```

freq-restormer > scripts > $ example.sh

```
#!/usr/bin/env bash
#SBATCH --time=47:59:59
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --partition=gpu_4 # or gpu_8 or gpu_4_a100 or gpu_4_h100
#SBATCH --gres=gpu:1
#SBATCH --mem=100G
#SBATCH --cpus-per-task=16
#SBATCH -J Name_of_JOB
#SBATCH --array=0-2%3
#SBATCH --output=slurm/Job_%A_%a.out
#SBATCH --mail-type=ALL
#SBATCH --mail-user=shashank.agnihotri@uni-mannheim.de


echo "Started at $(date)";
echo "Running job: $SLURM_JOB_NAME array id: $SLURM_ARRAY_TASK_ID using $SLURM_JOB_CPUS_PER_NODE cpus per node with given JID $SLURM_JOB_ID on queue $SLURM_JOB_PARTITION";


CONFIG="Motion_Deblurring/Options/Deblurring_Restormer.yml"

# python -W ignore basicsr/train.py -opt $CONFIG --flc --use_alpha --learn_alpha --blur --gpu_id 0 --zero_padding --upsample_method 'FreqAvgUp'

##### NNOOOWWW     NNOOOWWWWWW

if [[ $SLURM_ARRAY_TASK_ID -eq 0 ]]
then
    python -W ignore basicsr/train.py -opt $CONFIG --flc --adversarial --use_alpha --learn_alpha --blur --gpu_id 0 --zero_padding --upsample_method 'FreqAvgUp' --attack_method pgd --attack_iterations 3 --attack_alpha 0.01

elif [[ $SLURM_ARRAY_TASK_ID -eq 1 ]]
then
    python -W ignore basicsr/train.py -opt $CONFIG --flc --adversarial --use_alpha --learn_alpha --blur --gpu_id 0 --zero_padding --upsample_method 'SplitUp' --attack_method pgd --attack_iterations 3 --attack_alpha 0.01

elif [[ $SLURM_ARRAY_TASK_ID -eq 2 ]]
then
    python -W ignore basicsr/train.py -opt $CONFIG --flc --adversarial --use_alpha --drop_alpha --learn_alpha --blur --gpu_id 0 --zero_padding --upsample_method 'FreqAvgUp' --attack_method pgd --attack_iterations 3 --attack_alpha 0.01
else
    echo "All Submitted"
fi


end=`date +%s`
runtime=$((end-start))

echo Runtime: $runtime
```

Create a folder slurm/ inside the repository to store all slurm outputs

How to control jobs?

Job monitoring:
watch -n 1 squeue -u <user_name>

Cancelling a job:
Scancel job_id

Example:

```
(base) [ma_sagnihot@o05i14 ma_sagnihot-projects]$ squeue -u ma_sagnihot
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
   2543759_[0-9%9]    single Train_fr ma_sagni PD       0:00      1 (Priority)
         2525248_2    single Train_fr ma_sagni  R 2-01:33:37      1 o04c06
         2525248_1    single Train_fr ma_sagni  R 2-01:34:53      1 o07c02
         2525245_1    single Train_fr ma_sagni  R 2-03:23:56      1 p04c03
         2525248_0    single Train_fr ma_sagni  R 2-03:19:12      1 o07c01
         2525248_9    single Train_fr ma_sagni  R 1-17:09:03      1 p03c02
         2525248_6    single Train_fr ma_sagni  R 1-19:20:53      1 o04c02
         2525248_8    single Train_fr ma_sagni  R 1-18:07:15      1 p02c01
         2525248_5    single Train_fr ma_sagni  R 1-20:52:49      1 o04c05
         2525248_4    single Train_fr ma_sagni  R 1-22:15:09      1 p04c04
         2525248_3    single Train_fr ma_sagni  R 1-22:38:15      1 o07c05
         2525245_0    single Train_fr ma_sagni  R 2-05:03:36      1 p04c04
(base) [ma_sagnihot@o05i14 ma_sagnihot-projects]$ scancel 2543759
```

Example Project Structure

cospgd Public

Unpin | Unwatch 1 | Fork 0 | Star 8

main | 1 Branch | 0 Tags | Go to file t | Add file | Code

shashankskagnihotri Update README.md — c024843 · 5 months ago — 32 Commits

assets	editing readme	7 months ago
cospgd	editing readme	7 months ago
unet_backbones	editing readme	7 months ago
.gitignore	added attack_implementations to use for all subset repos	8 months ago
LICENSE.md	adding license	7 months ago
README.md	Update README.md	5 months ago
setup.py	better documentation	7 months ago

README | MIT license

CosPGD

About

The official repository for CosPGD: a unified white-box adversarial attack for pixel-wise prediction tasks.

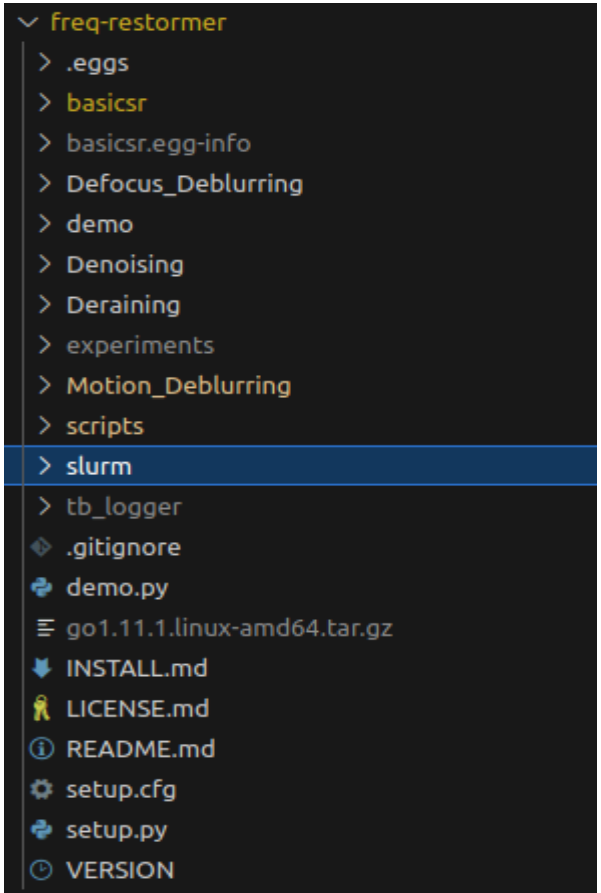
semantic, benchmarking, vision, segmentation, optical-flow, adversarial, attacks, semantic-segmentation, image-denoising, image-restoration, depth-estimation, image-deblurring, pixel-wise, unet-pytorch, unet-segmentation, benchmarking-functions, pixel-wise-regression, adversarial-att, pixel-wise-classification

Readme
MIT license
Activity
8 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Example sub project structure

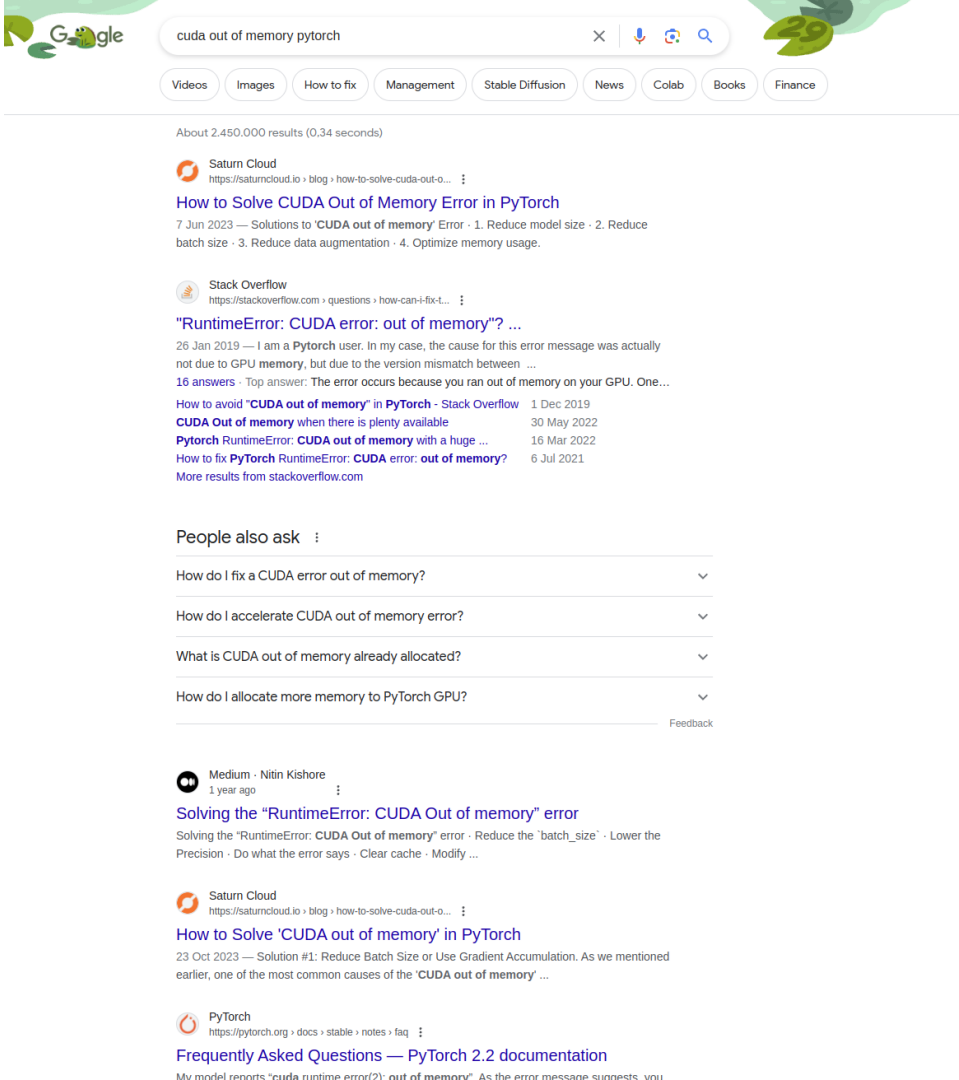


Google

Good website:

1. Stackoverflow
2. Pytorch.org
3. Python websites

The error message from others might be very different but the solution might be similar



APPENDIX: Pretraining

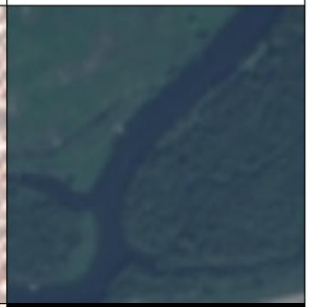
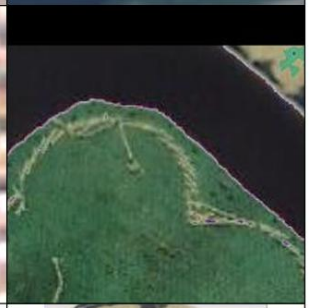
ImageNet pretraining: We use a learning rate of $2e^{-3}$ and train with a batch size of 1024 for 300 epochs.

CLIP pretraining: We train for 200K iterations on 0.7B image-text pairs from (Cao et al., 2023) using a learning rate of $5e^{-4}$ and a batch size of 65.5K.

DINO pretraining: We use a learning rate of $3e^{-3}$ and train for 100, 200, 300, 10K, 15K and 20K epochs on CC3M, Places, ImageNet, EuroSAT, ADE20K and HAM10K datasets, respectively, with a batch size of 1024.

Task-agnostic VFM feature distillation: We train for 100 epochs with a batch size of 2048 when using CC3M transfer set. We train for 200 epochs with a batch size of 2048 when using Places365 and ImageNet transfer sets. We train for 10K and 20K epochs with a batch size of 1024 when using ADE20K and HAM10K transfer sets, respectively. When using EuroSAT transfer set, we train for 10K epochs using a batch size of 2048. We use a learning rate $1e^{-3}$ for these distillation experiments.

Sneak Peak

Query Class	Highway	Residential	River
Query Image			
Matched Crop			
Retrieved Original Image			
Paired Text	Aviation Photos	Karen Metallic Cardigan	US Army Bullion Patch

Why this paper?

- Fairly well ablated recipe for distilling knowledge from VFMs to very Small models.
- Limited labels setting explored.
- Some very interesting insights into the “transfer dataset” used.

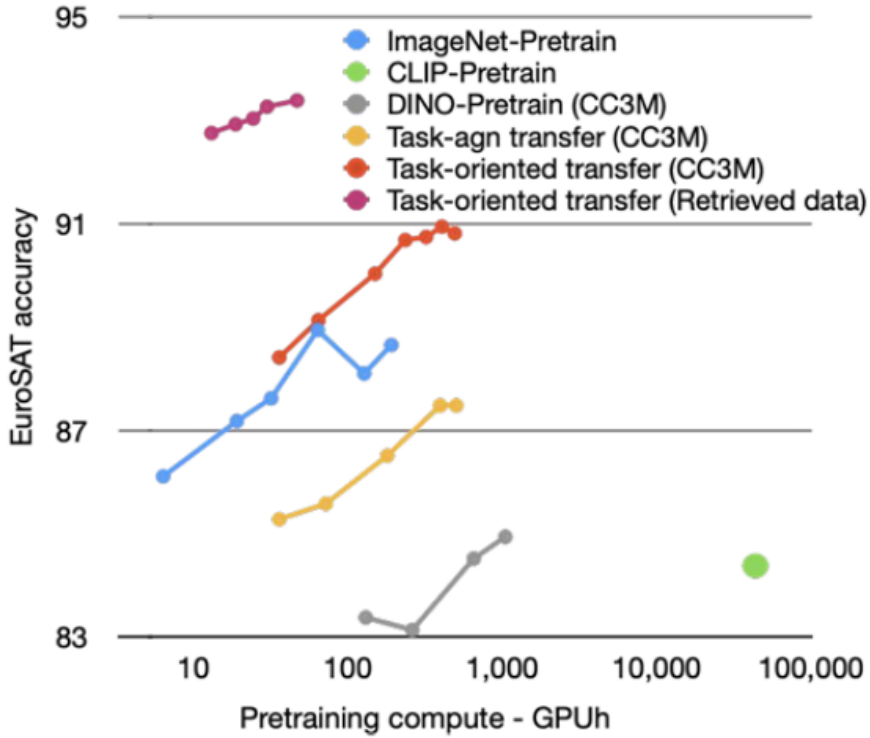
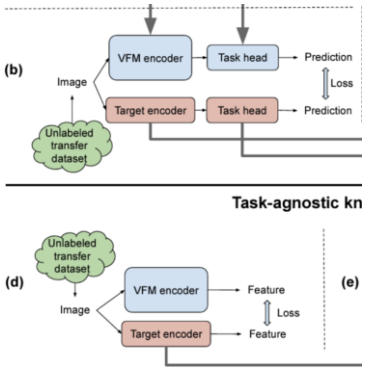


Figure 1

Task Oriented Transfer is the best!



Improvement over	When using generic CC3M transfer set					When using target task-related transfer set				
	HAM10K	EuroSAT	Places365	ImageNet	ADE20K	HAM10K	EuroSAT	Places365	ImageNet	ADE20K
Task-agnostic transfer	0.38 - 2.91	1.25 - 5.27	1.88 - 4.89	1.90 - **11.6**	3.41 - 10.5	1.01 - 3.53	0.71 - 2.35	1.38 - 3.47	3.40 - **6.30**	1.11 - 6.25
CLIP-Pretrain	2.89 - 6.79	1.81 - 9.55	2.20 - 5.50	2.30 - 11.2	6.52 - **20.9**	5.93 - 11.4	4.24 - 15.5	3.38 - 7.72	5.30 - **22.1**	5.45 - 19.2
IN-Pretrain	0.93 - 4.27	0.21 - 3.73	2.19 - 8.17	-	5.12 - **13.7**	3.97 - 8.93	2.64 - 9.64	3.37 - 9.89	-	3.53 - **12.0**
DINO-Pretrain	0.20 - 2.89	2.59 - 6.94	2.88 - 8.46	2.43 - **29.8**	7.86 - 15.4	4.70 - 8.76	5.57 - 12.6	2.72 - 4.50	4.00 - **21.7**	8.84 - 18.1

Table 1

Set A

VFMs used

1. DINOv2-ViT-L/14[1]
~300 million parameters

2. OpenCLIP-ViT-L/14[2]
~304.2 million parameters

Small Target Models used

1. FastVit-S12[3]
~8.8 million parameters

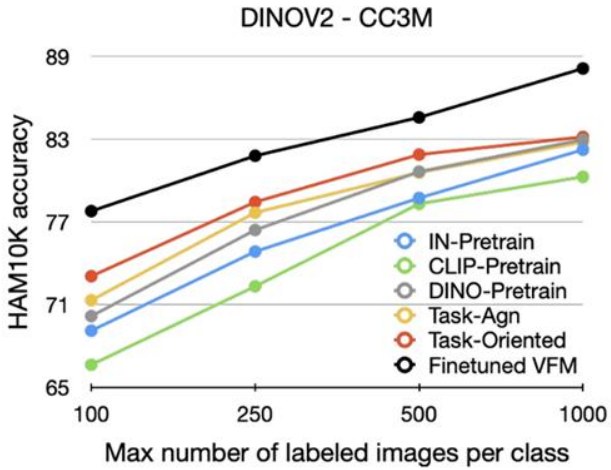
2. MobileVit-V2-1.0[4]
~2.3 million parameters

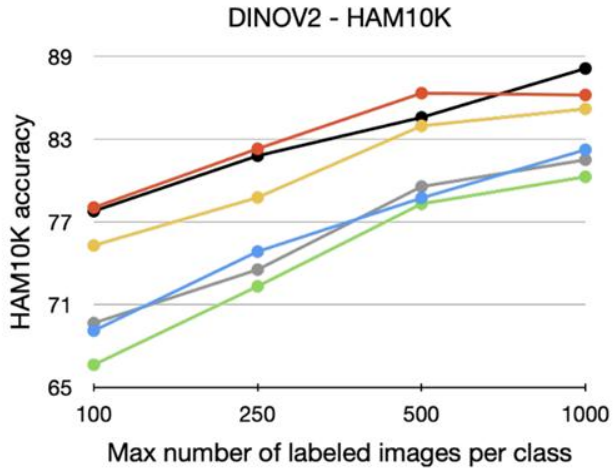
Example Set B

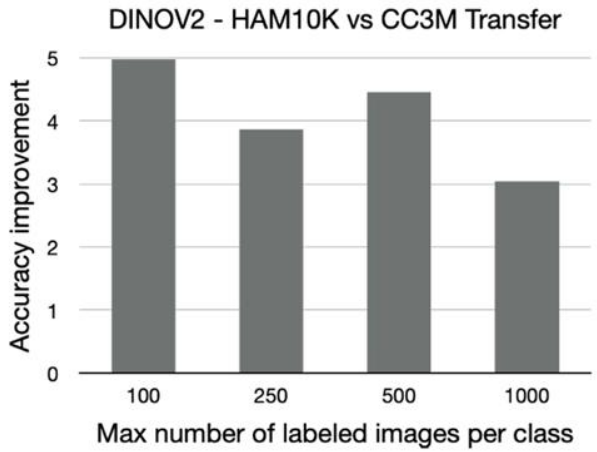
1. FastVit-S12[3]
~8.8 million parameters

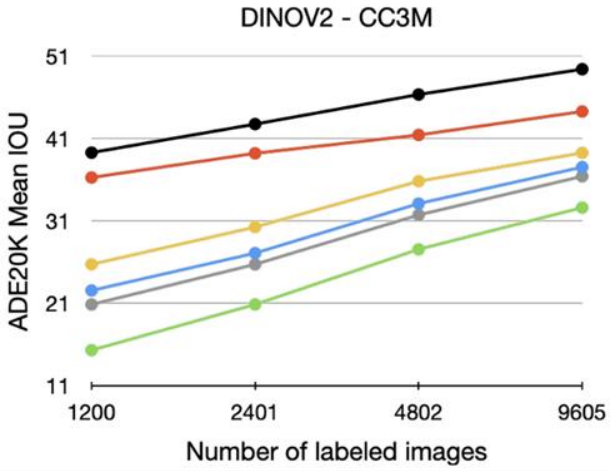
2. MobileVit-V2-1.0[4]
~2.3 million parameters

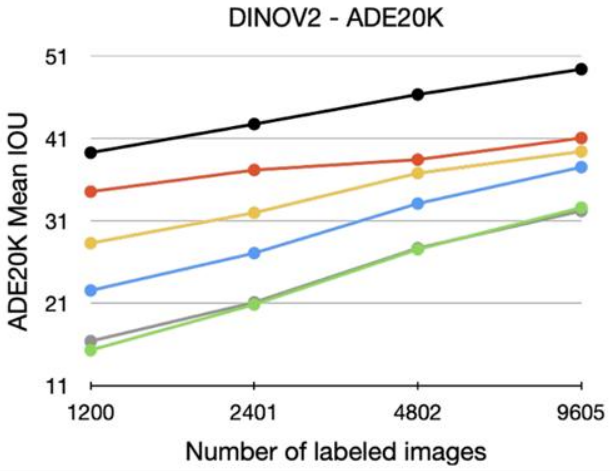
Task Oriented Transfer is the best!

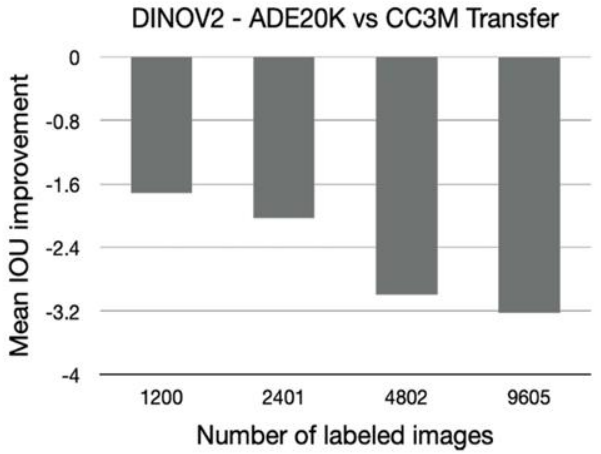




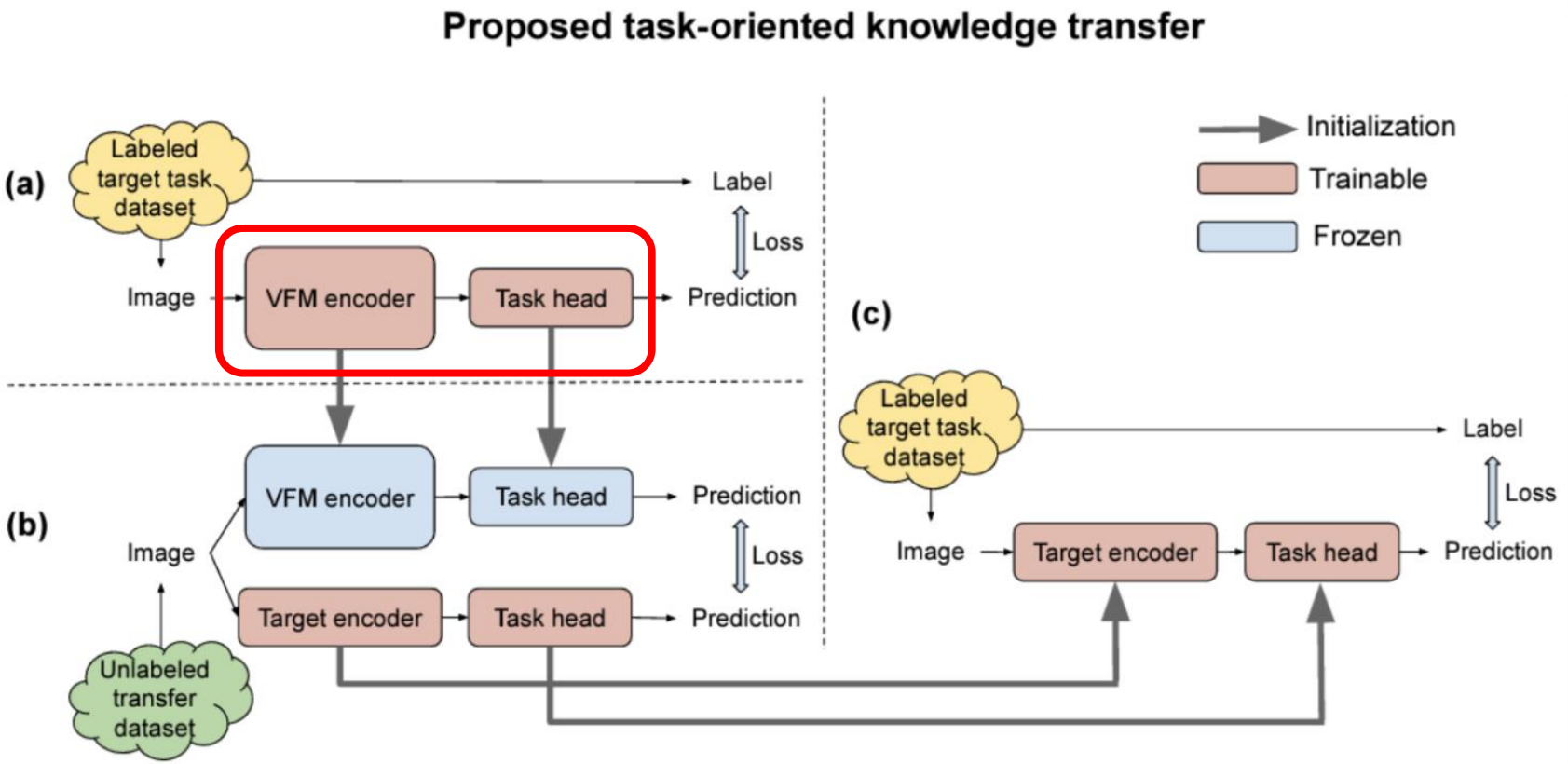








Ablating Task-Oriented Transfer

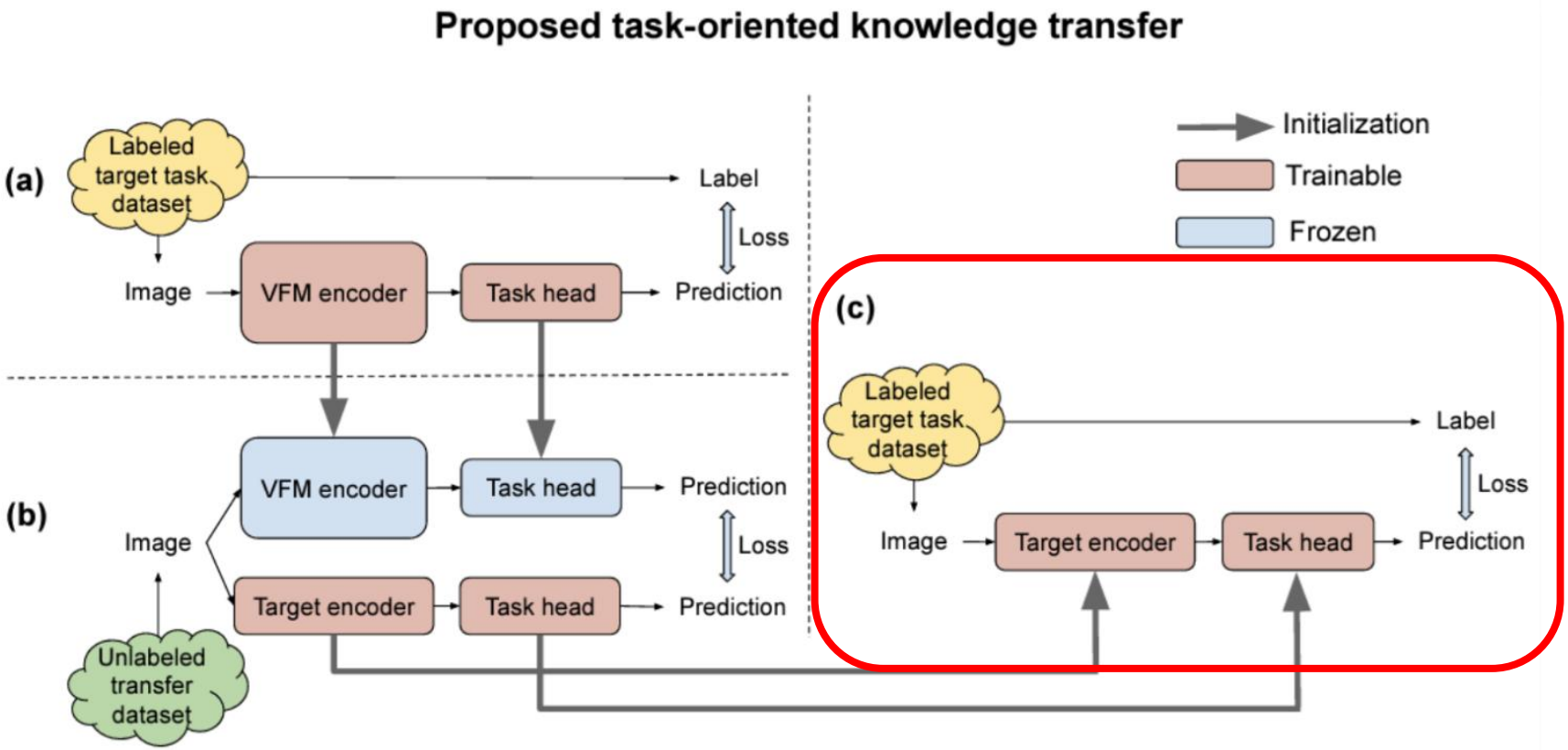


Ablating Task-Oriented Transfer: Fine-tuning v/s Linear Probing

Dataset	HAM10K	EuroSAT	Places365	ImageNet
Labeled data	100 img/cls	10 img/cls	250 img/cls	50%
VFM	DINOv2	OpenCLIP	OpenCLIP	OpenCLIP
VFM performance				
VFM-LP	71.10	87.33	52.39	84.07
VFM-FT	76.92	92.63	54.48	86.05
Task-oriented knowledge transfer with generic CC3M transfer set				
VFM-LP	69.97	89.59	52.87	78.13
VFM-FT	73.06	90.21	53.33	78.40
Task-oriented knowledge transfer with target task-related transfer set				
VFM-LP	74.60	94.37	54.69	81.01
VFM-FT	78.04	94.63	54.82	81.43

Table 2

Ablating Task-Oriented Transfer



Ablating Task-Oriented Transfer: Fine-tuning Target Model

Pretraining dataset - CC3M transfer set

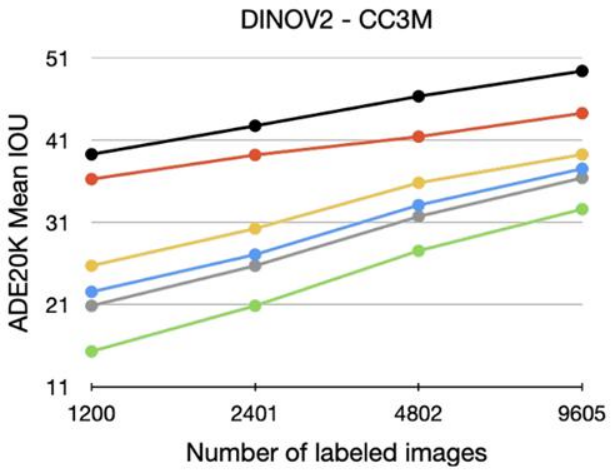
ADE20K	Number of labeled training images	1200	2401
	Performance gain	2.19	2.03
ImageNet	Percentage of labeled training images	10	25
	Performance gain	3.8	5.8
HAM10K	Maximum labeled training images per class	250	500
	Performance gain	9.99	7.41
EuroSAT	Number of labeled training images per class	5	10
	Performance gain	3.3	4.9
Places365	Number of labeled training images per class	250	1000
	Performance gain	0.47	0.76

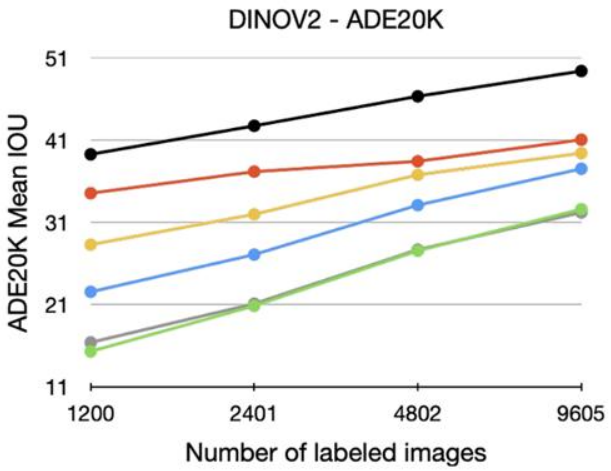
Pretraining dataset - Target task-related transfer set

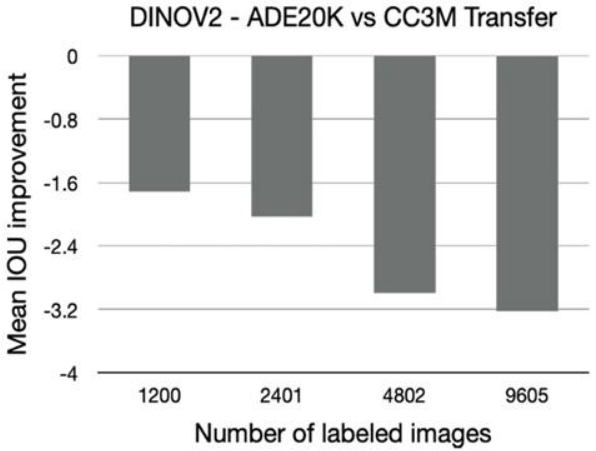
ADE20K	Number of labeled training images	1200	2401
	Performance gain	1.23	1.25

Table 3

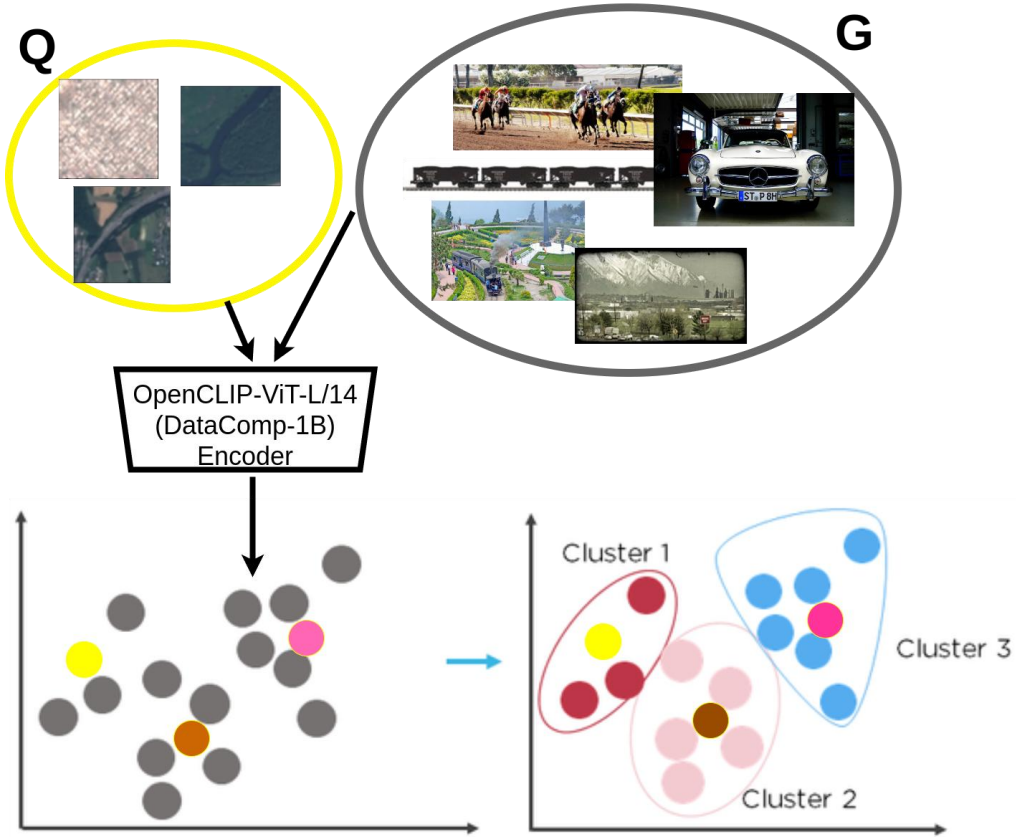
Retrieval Augmented Knowledge Transfer

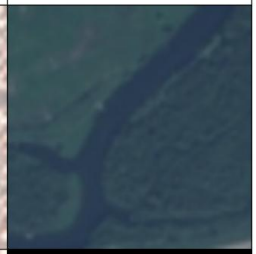
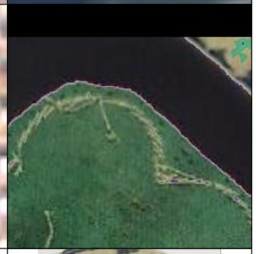






Retrieval Augmented Knowledge Transfer



Query Class	Highway	Residential	River
Query Image			
Matched Crop			
Retrieved Original Image			
Paired Text	Aviation Photos	Karen Metallic Cardigan	US Army Bullion Patch

Retrieval Augmented Knowledge Transfer

ADE20K mean IOU (transfer from DINOv2 VFM)

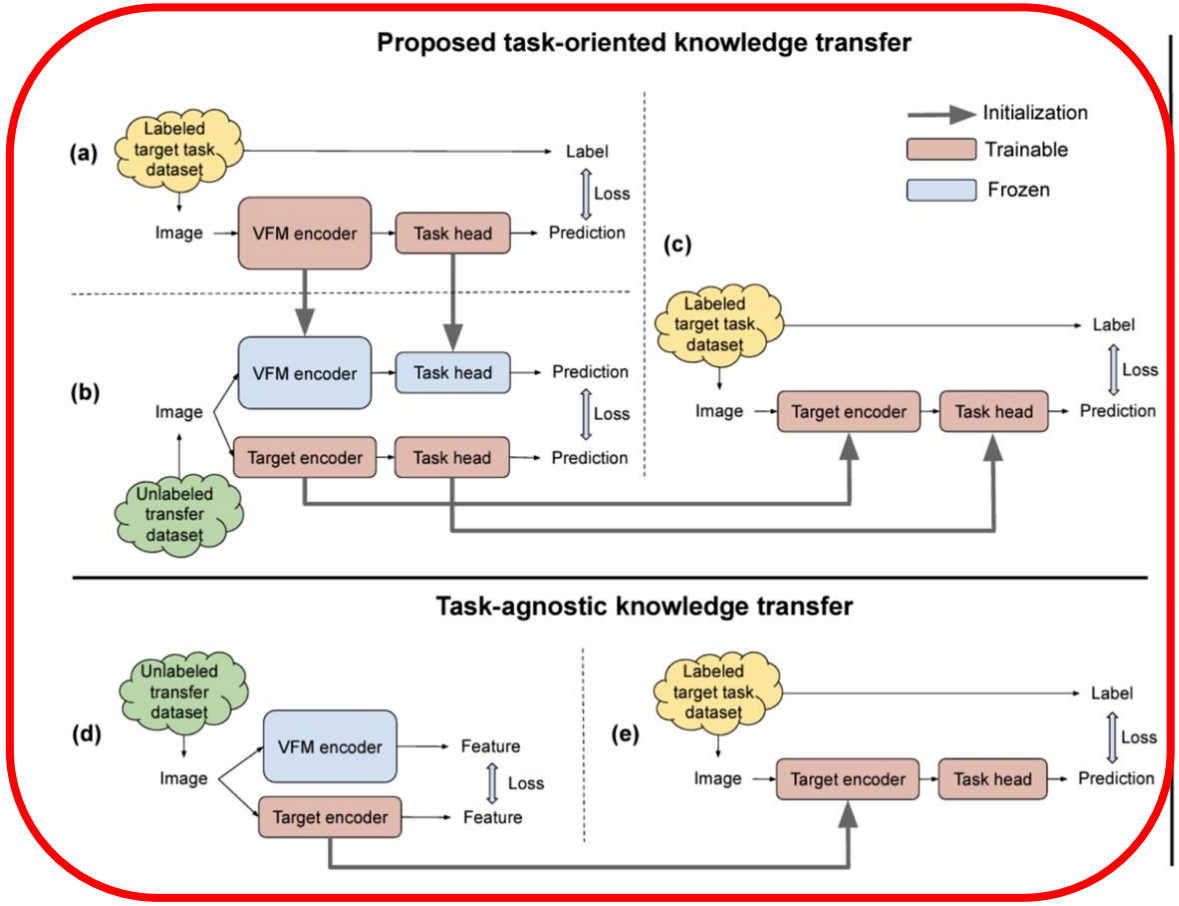
Labeled images / Query set size	1200	2401	4802	9605
Full ADE20K transfer (19.2K)	34.57	37.19	38.45	41.07
CC3M transfer (2.87M)	36.28	39.22	41.44	44.29
Retrieval augmented transfer (154K)	37.65	40.40	43.28	44.93

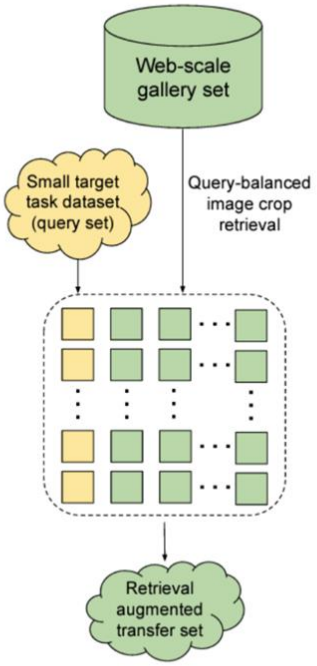
EuroSAT accuracy (transfer from OpenCLIP VFM)

Labeled images / Query set size	50	100	250
Full EuroSAT transfer (2.7K)	90.74	94.63	96.83
CC3M transfer (2.87M)	85.14	90.21	94.25
Retrieval augmented transfer (51K)	89.23	93.25	96.35

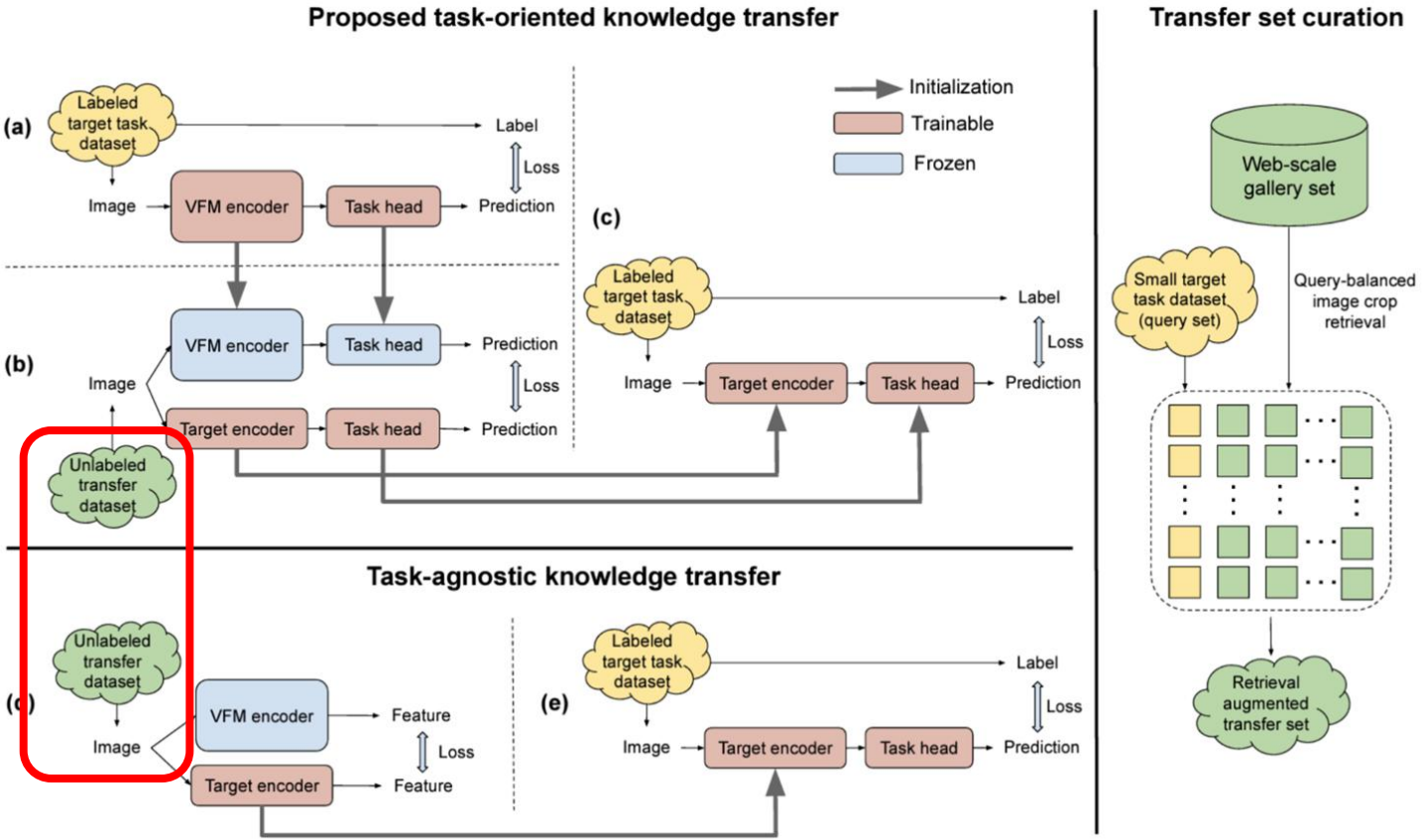
Table 4. Comparison between various transfer sets in terms of their effectiveness for task-oriented knowledge transfer.

Conclusion

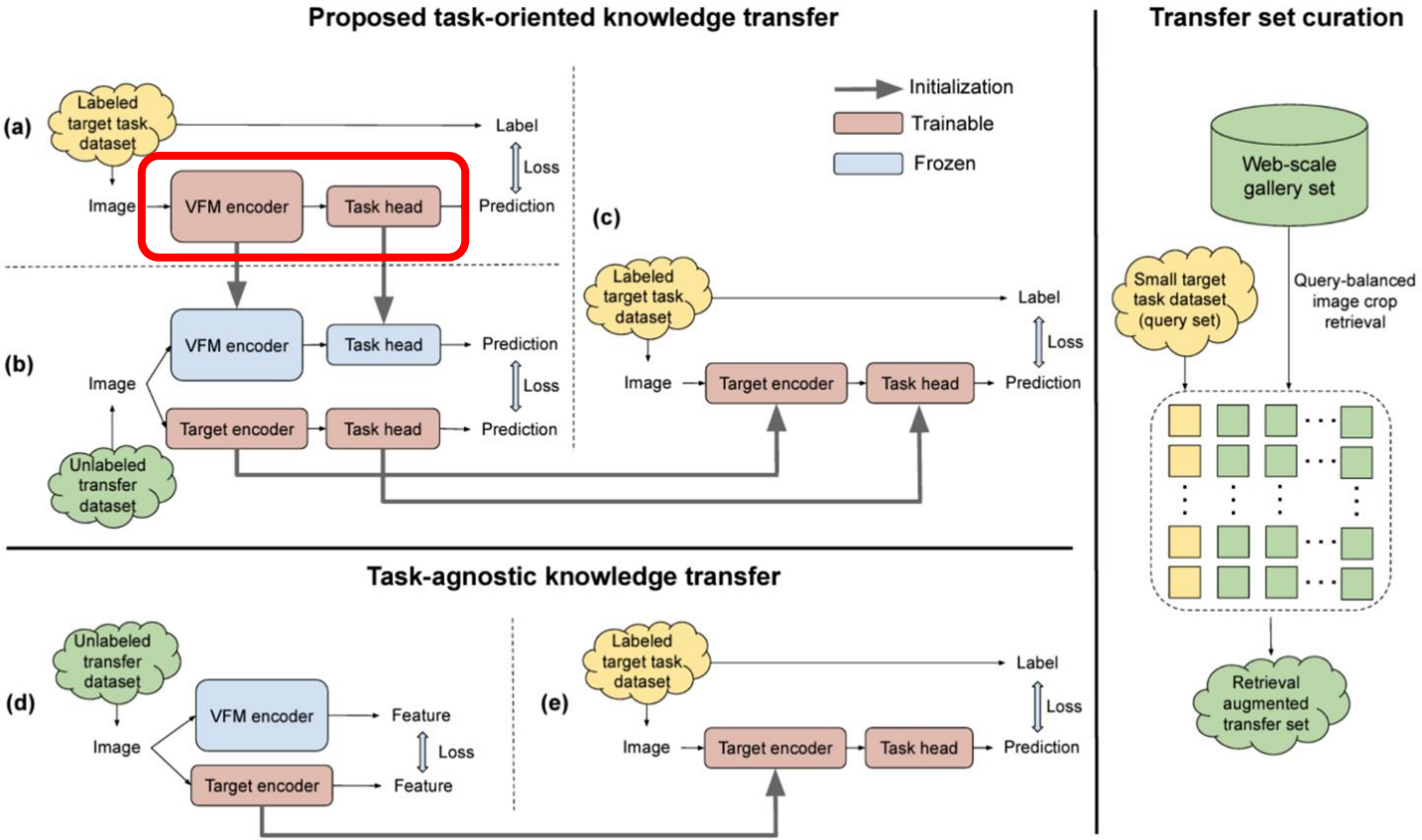

Proposed task-oriented knowledge transfer
(a)
Labeled target task dataset
Label
Loss
Image
VFM encoder
Task head
Prediction
(b)
VFM encoder
Task head
Prediction
Image
Loss
Target encoder
Task head
Prediction
Unlabeled transfer dataset
Initialization
Trainable
Frozen
(c)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Task-agnostic knowledge transfer
(d)
Unlabeled transfer dataset
VFM encoder
Feature
Image
Loss
Target encoder
Feature
(e)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction


Transfer set curation
Web-scale gallery set
Small target task dataset (query set)
Query-balanced image crop retrieval
Retrieval augmented transfer set

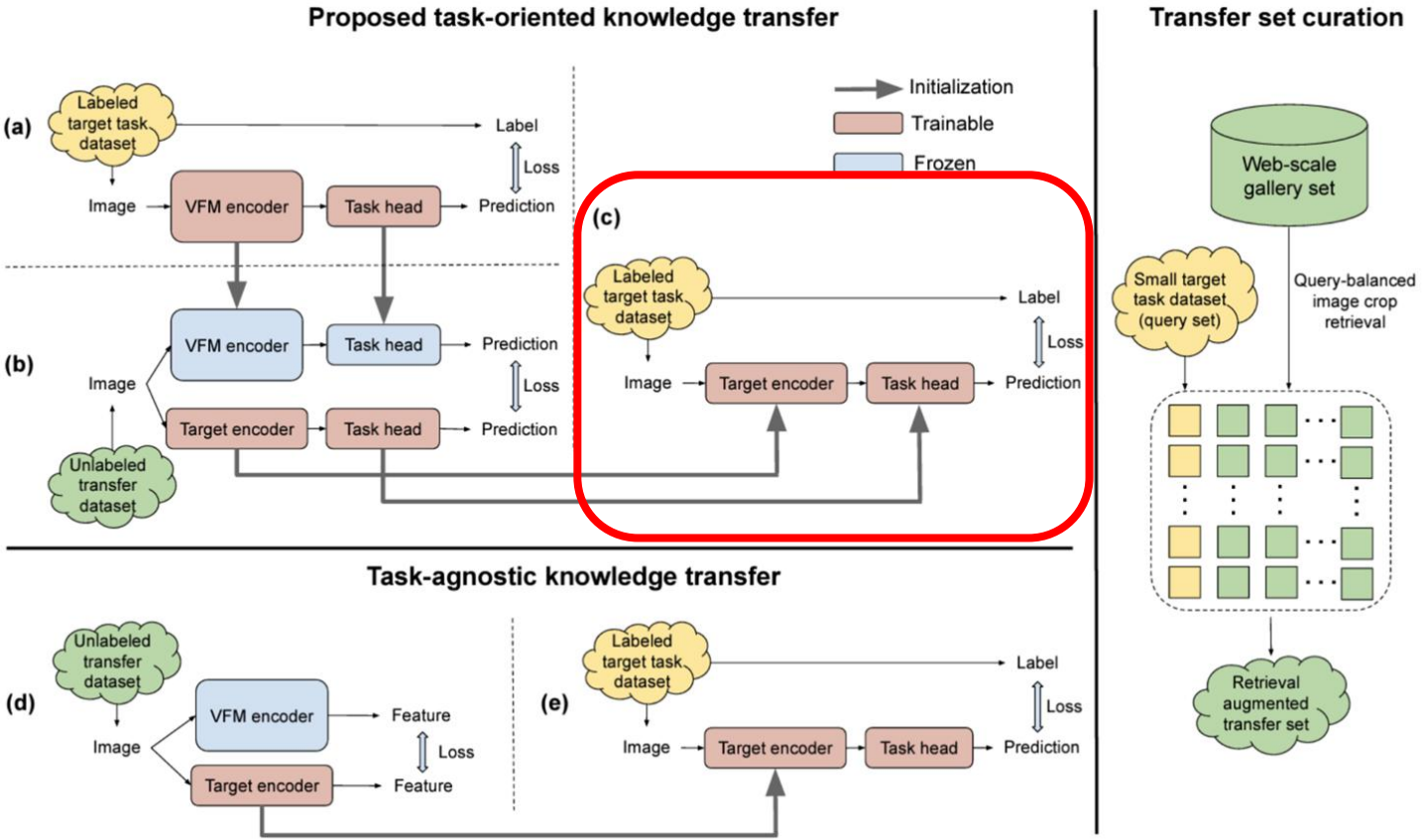
Conclusion


Proposed task-oriented knowledge transfer
Transfer set curation
Initialization
Trainable
Frozen
(a)
Labeled target task dataset
Label
Loss
Image
VFM encoder
Task head
Prediction
(b)
VFM encoder
Task head
Prediction
Loss
Image
Target encoder
Task head
Prediction
Unlabeled transfer dataset
(c)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Task-agnostic knowledge transfer
(d)
Unlabeled transfer dataset
Image
VFM encoder
Feature
Loss
Target encoder
Feature
(e)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Web-scale gallery set
Small target task dataset (query set)
Query-balanced image crop retrieval
Retrieval augmented transfer set

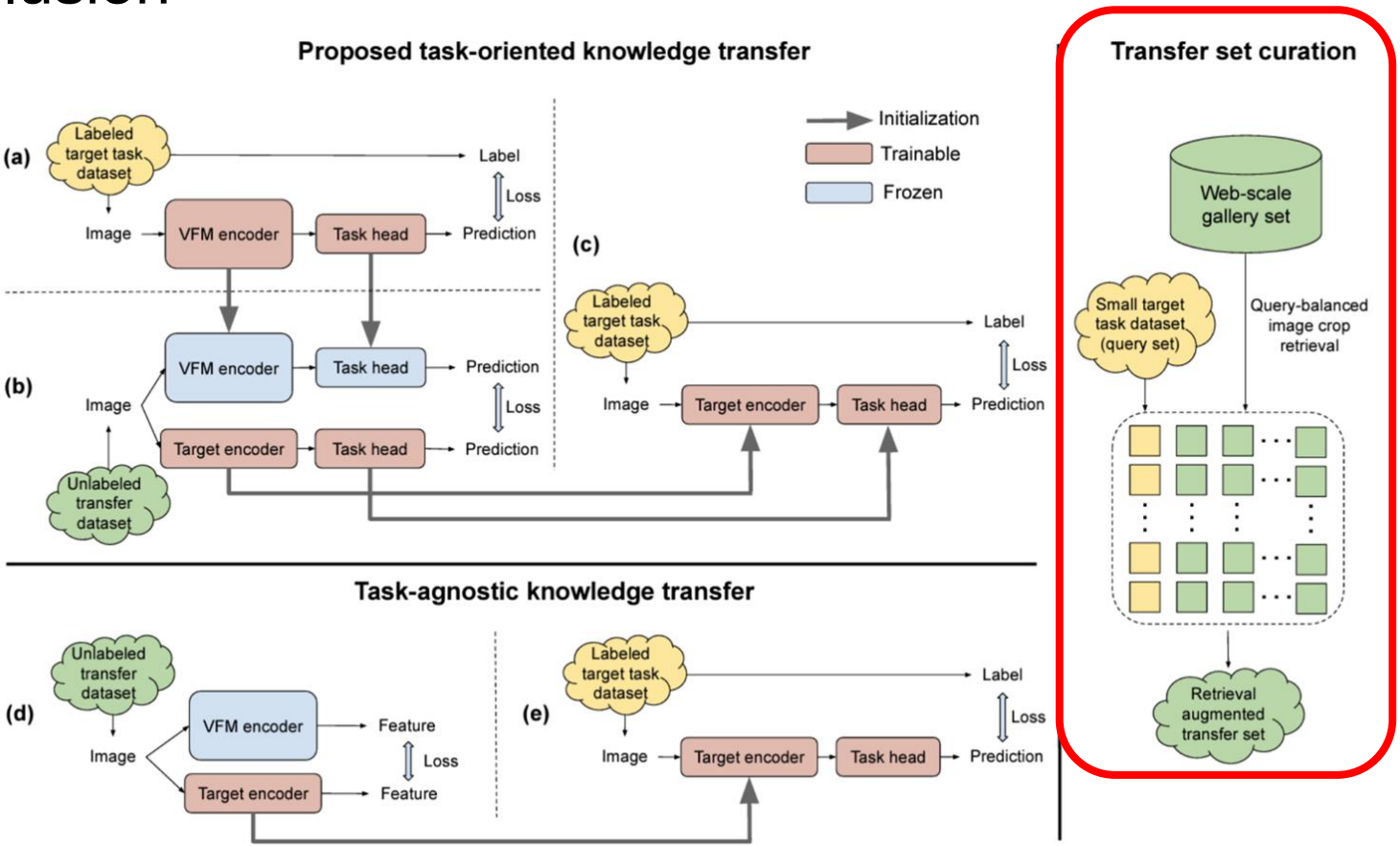
Conclusion


Proposed task-oriented knowledge transfer
Transfer set curation
Initialization
Trainable
Frozen
(a)
Labeled target task dataset
Label
Loss
Image
VFM encoder
Task head
Prediction
(b)
VFM encoder
Task head
Prediction
Loss
Image
Target encoder
Task head
Prediction
Unlabeled transfer dataset
(c)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Task-agnostic knowledge transfer
(d)
Unlabeled transfer dataset
VFM encoder
Feature
Loss
Image
Target encoder
Feature
(e)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Web-scale gallery set
Small target task dataset (query set)
Query-balanced image crop retrieval
Retrieval augmented transfer set

Conclusion


Proposed task-oriented knowledge transfer
Transfer set curation
Initialization
Trainable
Frozen
(a)
Labeled target task dataset
Label
Loss
Image
VFM encoder
Task head
Prediction
(b)
VFM encoder
Task head
Prediction
Loss
Image
Target encoder
Task head
Prediction
Unlabeled transfer dataset
(c)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Task-agnostic knowledge transfer
(d)
Unlabeled transfer dataset
VFM encoder
Feature
Loss
Image
Target encoder
Feature
(e)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Web-scale gallery set
Small target task dataset (query set)
Query-balanced image crop retrieval
Retrieval augmented transfer set

Conclusion


Proposed task-oriented knowledge transfer
Initialization
Trainable
Frozen
(a)
Labeled target task dataset
Label
Loss
Image
VFM encoder
Task head
Prediction
(b)
VFM encoder
Task head
Prediction
Image
Loss
Target encoder
Task head
Prediction
Unlabeled transfer dataset
(c)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Task-agnostic knowledge transfer
(d)
Unlabeled transfer dataset
VFM encoder
Feature
Image
Loss
Target encoder
Feature
(e)
Labeled target task dataset
Label
Loss
Image
Target encoder
Task head
Prediction
Transfer set curation
Web-scale gallery set
Small target task dataset (query set)
Query-balanced image crop retrieval
Retrieval augmented transfer set

Finetuned VFM (DINOv2)		87.36	92.65	95.66
Finetuned VFM (OpenCLIP)		87.99	92.63	96.08
IN-Pretrain		81.41	88.65	94.04
CLIP-Pretrain		75.59	84.38	92.44
DINO-Pretrain (CC3M)		78.20	84.83	91.66
DINO-Pretrain (EuroSAT)		78.50	83.65	91.11
DINOv2 (CC3M transfer)	Task-Agn (Patch)	76.52	86.11	92.66
	Task-Agn (Image)	79.81	87.66	93.52
	Task-Oriented	83.72	90.75	94.77
DINOv2 (EuroSAT transfer)	Task-Agn (Patch)	90.23	93.34	95.97
	Task-Agn (Image)	86.54	89.56	94.19
	Task-Oriented	91.05	94.71	96.68
OpenCLIP (CC3M transfer)	Task-Agn	79.87	87.10	92.91
	Task-Oriented	85.14	90.21	94.25
OpenCLIP (EuroSAT transfer)	Task-Agn	88.57	92.28	95.5
	Task-Oriented	90.74	94.63	96.83

Table 6. EuroSAT classification accuracy for FastViT target model.

Labeled training images per class		50	250	1000
Finetuned VFM (DINOv2)		47.56 ± 0.02	54.11 ± 0.27	56.95 ± 0.11
Finetuned VFM (OpenCLIP)		48.30 ± 0.59	54.26 ± 0.19	57.03 ± 0.17
IN-Pretrain		40.58 ± 0.14	47.96 ± 0.09	52.33 ± 0.14
CLIP-Pretrain		$45.17 \pm 0.03^*$	47.83 ± 0.30	52.37 ± 0.47
DINO-Pretrain (CC3M)		40.29 ± 0.26	48.20 ± 0.16	52.74 ± 0.12
DINO-Pretrain (Places365)		45.97 ± 0.18	50.85 ± 0.12	53.96 ± 0.06
DINOv2 (CC3M transfer)	Task-Agn (Patch)	$42.46 \pm 0.02^*$	49.60 ± 0.11	53.49 ± 0.06
	Task-Agn (Image)	$44.52 \pm 0.13^*$	49.75 ± 0.07	53.45 ± 0.04
	Task-Oriented	47.81 ± 0.05	53.09 ± 0.05	55.62 ± 0.05
DINOv2 (Places365 transfer)	Task-Agn (Patch)	$46.45 \pm 0.05^*$	51.26 ± 0.20	54.43 ± 0.03
	Task-Agn (Image)	$47.76 \pm 0.04^*$	51.45 ± 0.32	54.45 ± 0.21
	Task-Oriented	49.14 ± 0.02	54.51 ± 0.05	56.68 ± 0.01
OpenCLIP (CC3M transfer)	Task-Agn	$45.44 \pm 0.03^*$	49.56 ± 0.05	53.59 ± 0.09
	Task-Oriented	48.75 ± 0.01	53.33 ± 0.05	55.75 ± 0.06
OpenCLIP (Places365 transfer)	Task-Agn	$48.83 \pm 0.07^*$	51.92 ± 0.26	54.74 ± 0.10
	Task-Oriented	50.47 ± 0.01	54.82 ± 0.04	56.80 ± 0.02

Table 7. Places365 classification accuracy for FastViT target model. The results marked with * are obtained by training only the classification layer instead of the entire model in the finetuning stage. Full finetuning produced inferior results in these cases.

Labeled training images		1200	2401	4802	9605
Finetuned VFM (DINOv2)		39.32 ± 0.11	42.76 ± 0.08	46.35 ± 0.09	49.42 ± 0.14
IN-Pretrain		22.58 ± 0.07	27.11 ± 0.06	33.12 ± 0.25	37.54 ± 0.02
CLIP-Pretrain		15.34 ± 0.20	20.88 ± 0.26	27.57 ± 0.11	32.63 ± 0.08
DINO-Pretrain (CC3M)		20.88 ± 0.27	25.75 ± 0.25	31.74 ± 0.08	36.43 ± 0.02
DINO-Pretrain (ADE20K)		16.43 ± 0.09	21.14 ± 0.20	27.71 ± 0.23	32.23 ± 0.23
DINOv2 (CC3M transfer)	Task-Agn (Patch)	25.75 ± 0.55	30.27 ± 0.14	35.83 ± 0.11	39.28 ± 0.54
	Task-Oriented	36.28 ± 0.01	39.22 ± 0.19	41.44 ± 0.15	44.29 ± 0.03
DINOv2 (ADE20K transfer)	Task-Agn (Patch)	28.32 ± 0.46	32.0 ± 0.19	36.81 ± 0.08	39.44 ± 0.22
	Task-Oriented	34.57 ± 0.02	37.19 ± 0.10	38.45 ± 0.05	41.07 ± 0.21

Table 8. ADE20K mean IOU for FastViT target model.

Maximum labeled training images per class		100	250	500	1000
Finetuned VFM (DINOv2)		77.78 ± 0.61	81.79 ± 0.97	84.57 ± 1.88	88.12 ± 0.56
Finetuned VFM (OpenCLIP)		75.13 ± 0.82	79.70 ± 1.19	85.12 ± 0.35	86.38 ± 0.74
IN-Pretrain		69.11 ± 1.56	74.85 ± 0.82	78.73 ± 1.88	82.23 ± 1.37
CLIP-Pretrain		66.64 ± 1.75	72.33 ± 1.11	78.31 ± 1.99	80.27 ± 2.08
DINO-Pretrain (CC3M)		70.17 ± 1.60	76.41 ± 1.00	80.64 ± 0.76	82.96 ± 0.31
DINO-Pretrain (HAM10K)		69.66 ± 2.52	73.54 ± 1.86	79.56 ± 0.81	81.50 ± 1.46
DINOv2 (CC3M transfer)	Task-Agn (Patch)	71.76 ± 2.28	75.68 ± 1.83	80.34 ± 0.74	80.95 ± 1.16
	Task-Agn (Image)	71.32 ± 1.94	77.67 ± 1.48	80.58 ± 0.53	82.78 ± 1.01
	Task-Oriented	73.06 ± 0.79	78.44 ± 0.41	81.88 ± 0.64	83.16 ± 0.83
DINOv2 (HAM10K transfer)	Task-Agn (Patch)	75.29 ± 1.22	78.77 ± 1.55	83.96 ± 0.10	85.19 ± 0.40
	Task-Agn (Image)	72.18 ± 2.57	76.85 ± 0.74	81.68 ± 0.85	83.64 ± 0.59
	Task-Oriented	78.04 ± 0.05	82.30 ± 0.14	86.33 ± 0.16	86.20 ± 0.32
OpenCLIP (CC3M transfer)	Task-Agn	70.77 ± 1.00	78.40 ± 0.77	81.26 ± 0.37	81.55 ± 1.31
	Task-Oriented	72.53 ± 0.63	79.12 ± 1.27	82.56 ± 0.4	84.46 ± 0.65
OpenCLIP (HAM10K transfer)	Task-Agn	73.81 ± 0.44	78.40 ± 0.27	81.81 ± 0.82	84.24 ± 1.53
	Task-Oriented	75.04 ± 0.06	81.15 ± 0.01	84.70 ± 0.08	87.39 ± 0.08

Table 9. HAM10K classification accuracy for FastViT target model.