TEXAS A&M UNIVERSITY

NEURAL INFORMATION PROCESSING SYSTEMS

Tune-Up Open-Weight CLIP: Optimization Framework for Self-Supervised Fine-tuning of CLIP

Anant Mehta Xingyu Chen Xiyuan Wei Tianbao Yang



Spotlight Poster

Code: https://github.com/Optimization-AI/TuneCLIP

Introduction

- Recent CLIP improvements rely on large-scale pretraining from scratch using ever-larger **datasets**, new **objectives** or **optimization algorithms**, which require ***billions*** of image-text pairs, massive GPU clusters, weeks of computation

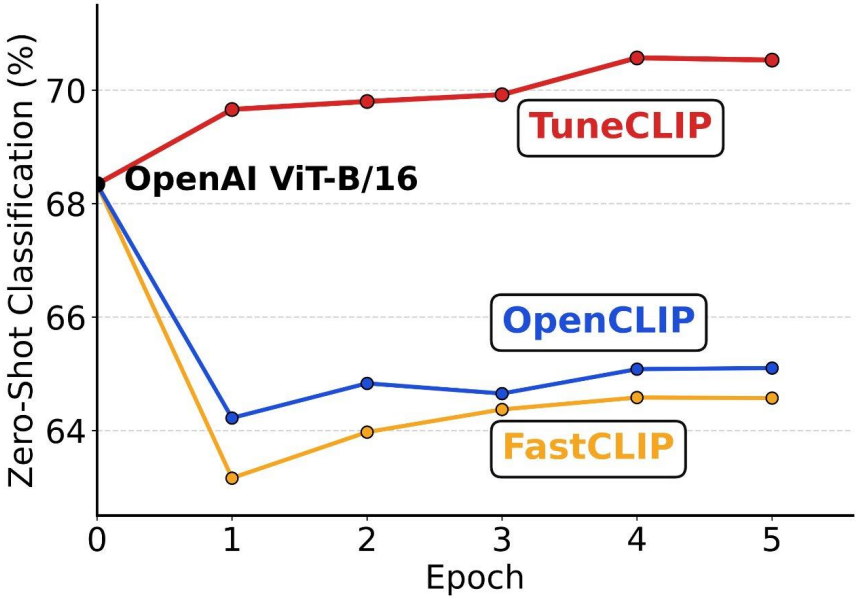
- OpenAI reports using 592 V100 GPUs (over 12 days) for CLIP training !!

- Supervised fine-tuning improves in-domain performance but often degrades generalization under distribution shifts, limiting its ability to improve CLIP as a general-purpose model

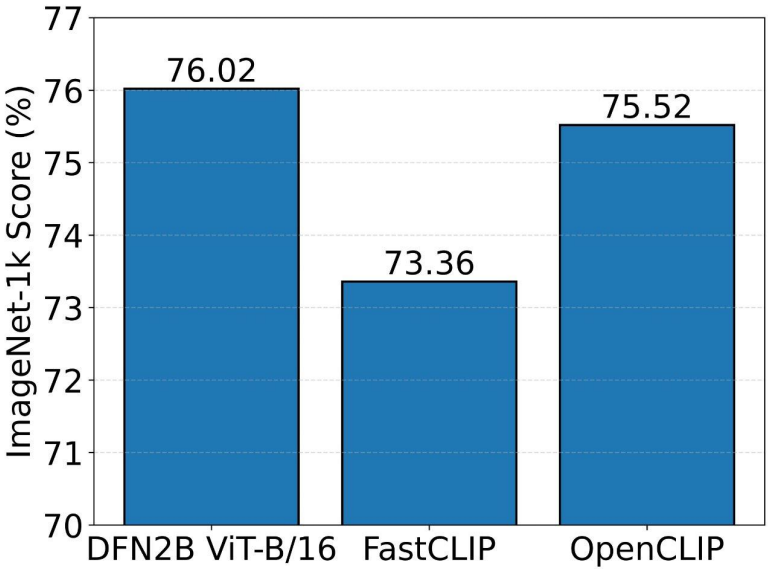
- **Self-supervised fine-tuning (SSFT)**: improves a pretrained CLIP model's overall representational quality and general-purpose performance, rather than adapting it to a specific downstream task

  Pretraining-style fine-tuning

Key Challenges & Observations



Sharp performance drop at the onset of fine-tuning



Even with its own pretraining data

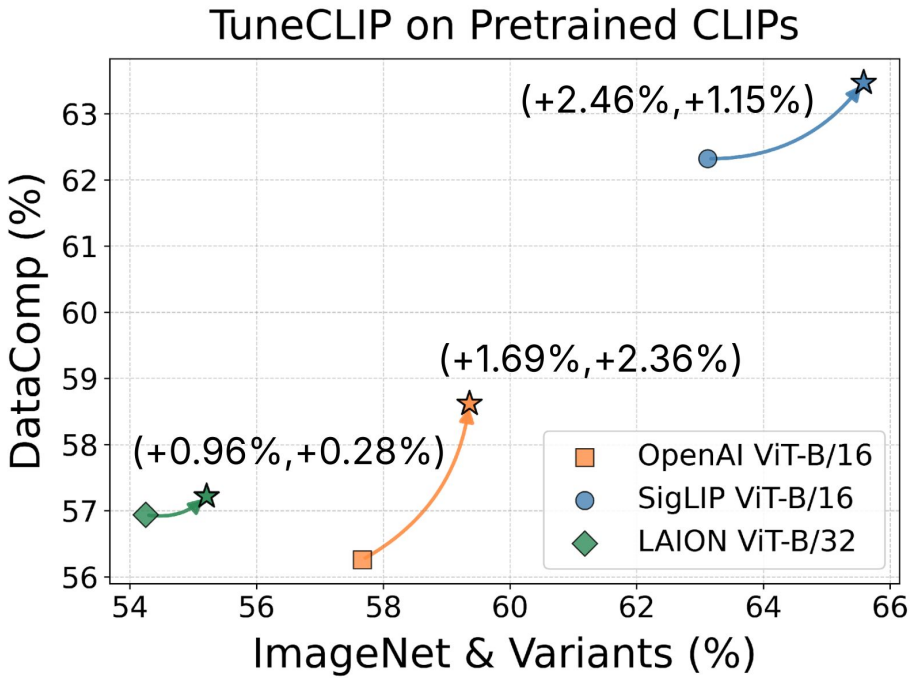
Two Causes

- **Optimization Perspective:**
  Contrastive losses lack unbiased stochastic gradient estimators , making optimization highly sensitive at initialization and causing sharp performance drops at the start of training

  We call this cold-start bias

- **Learning Perspective:**
  Semantically similar samples are mistakenly treated as negatives; fine-tuning amplifies this effect, distorting embeddings and degrading retrieval performance

TuneCLIP: A two-stage Solution



Stage I: Optimizer Statistics Recovery (OSR)

$$\mathcal{L}_{\mathrm{GCL}}(\omega) = \frac{\tau}{n} \sum_{i=1}^{n} \Big[\log\left(\varepsilon + \Phi_1(\omega, i, \mathcal{D})\right) + \log\left(\varepsilon + \Phi_2(\omega, i, \mathcal{D})\right) \Big]$$

- $\Phi_*(\omega, i, \mathcal{D}) = \dfrac{1}{|\mathcal{D}|} \displaystyle\sum_{j \in \mathcal{D}} \exp\left(\dfrac{s_{ij} - s_{ii}}{\tau}\right)$

$$u_{i,*}^{(t)} = (1 - \gamma_t)\, u_{i,*}^{(t-1)} + \gamma_t\, \Phi_*(\omega_{t-1}, i, \mathcal{B}_t)$$

- $u_{i,*} \approx \Phi_*(\omega, i, \mathcal{D})$
- $\mathcal{B}_t \subset \mathcal{D}$: mini-batch

We optimize a **Global Contrastive Loss** which is a development over standard **Mini-Batch Contrastive Loss**

Stage I: Optimizer Statistics Recovery (OSR)

$$T = O\left(\frac{n}{B^2\epsilon^4}\left(\Delta_0 + \frac{B}{n}M_0 + U_{x,0} + U_{z,0}\right)\right)$$

- $\Delta_0 = \mathcal{L}_{\mathrm{GCL}}(\omega_0) - \min_\omega \mathcal{L}_{\mathrm{GCL}}(\omega)$
- $M_0 = \|m_0 - \nabla_\omega \mathcal{L}_{\mathrm{GCL}}(\omega_0)\|^2$
- $U_{x,0} = \frac{1}{2n}\|u_x^{(0)} - \Phi_1(\omega_0, \mathcal{D})\|^2$
- $U_{z,0} = \frac{1}{2n}\|u_z^{(0)} - \Phi_2(\omega_0, \mathcal{D})\|^2$

The convergence is proportional to **initial erro**r of model parameters, momentum and inner function estimator !!

Algorithm 1: Optimizer Statistics Recovery (OSR)

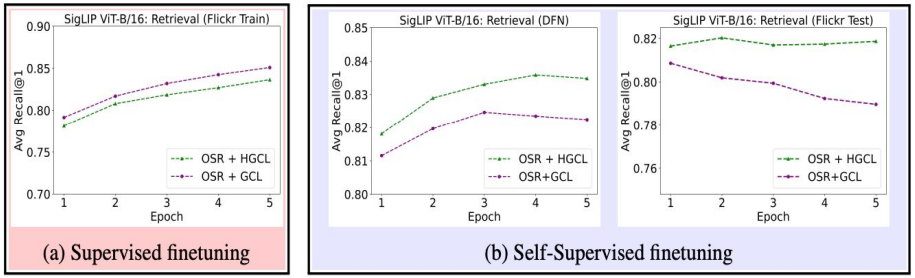
Init: $\boldsymbol{\omega_0}$ (Pretrained), $m_0 \leftarrow [0]^{|\mathcal{M}|}$, $v_0 \leftarrow [0]^{|\mathcal{M}|}$, $u_x^{(0)} \leftarrow [0]^{|\mathcal{D}|}$, $u_z^{(0)} \leftarrow [0]^{|\mathcal{D}|}$
for *iteration* $t = 1$ **to** T **do**

Sample $\mathcal{B}_t \subset \mathcal{D}$	`// mini-batch sampling`
$u_{i,x}^{(t)} \leftarrow (1-\gamma_t)u_{i,x}^{(t-1)} + \gamma_t\, \Phi_1(\boldsymbol{\omega_0}, i, \mathcal{B}_t), \forall i \in \mathcal{B}_t$	`// refer (3)`
$u_{i,z}^{(t)} \leftarrow (1-\gamma_t)u_{i,z}^{(t-1)} + \gamma_t\, \Phi_2(\boldsymbol{\omega_0}, i, \mathcal{B}_t), \forall i \in \mathcal{B}_t$	
Compute $g_t = G(\boldsymbol{\omega_0}, \mathcal{B}_t)$	`// frozen` $\boldsymbol{\omega_0}$
Update $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1)g_t$	`// first moment`
Update $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2)(g_t \odot g_t)$	`// second moment`

Return : $m^\star \leftarrow m_T,\ v^\star \leftarrow v_T,\ u^\star \leftarrow \{u_{i,x}^{(T)}, u_{i,z}^{(T)}\}_{i\in\mathcal{D}}$

Let’s Recover them..

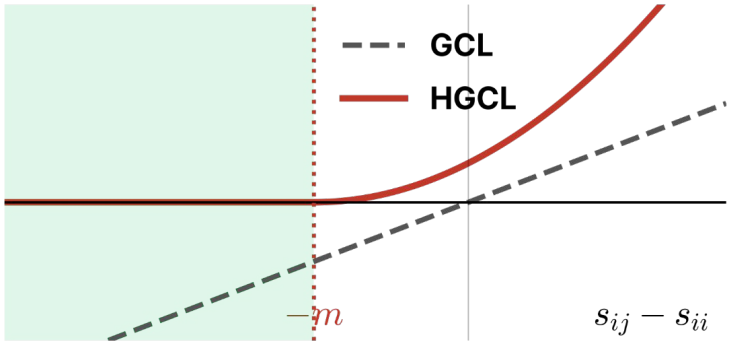
Stage II: Hinged Global Contrastive Loss (HGCL)

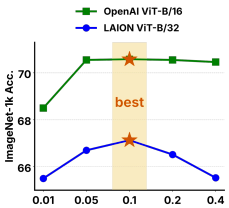


With OSR, fine-tuning stabilizes and leads to strong improvements initially.

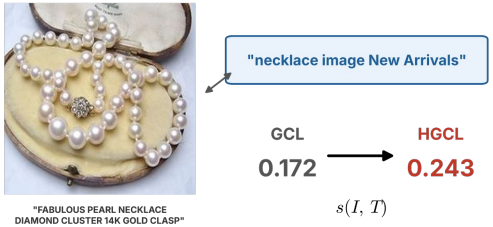
However, retrieval performance still decline as fine-tuning progresses

$$\ell_{ij} \propto s_{ij} - s_{ii} \longrightarrow \ell_{ij} = [\, s_{ij} - s_{ii} + m \,]_+^2$$





We attribute this generalization gap to the **over-penalization of false negatives** in web-scale datasets



Main Results

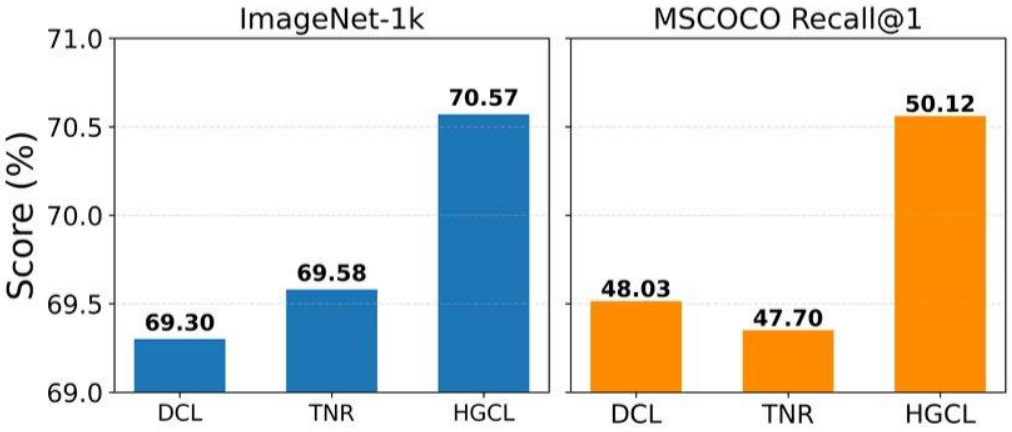
Base Model	Method	Wall C.T. (hrs)	IN & Variants	Retrieval	DataComp
OpenAI ViT-B/16	Baseline	N/A	57.67	57.46	56.26
	FastCLIP	4.21	54.57 (↓)	51.88 (↓)	53.53 (↓)
	OpenCLIP	5.46	54.99 (↓)	57.81	55.11 (↓)
	CLIP-Ref.	1.01	55.96(↓)	58.82	56.18(↓)
	TuneCLIP	**8.62**	**59.36** (+1.69)	**64.12** (+6.66)	**58.62** (+2.36)
SigLIP ViT-B/16	Baseline	N/A	63.12	69.32	62.32
	FastCLIP	4.28	39.22 (↓)	43.37 (↓)	45.80 (↓)
	OpenCLIP	7.55	40.21 (↓)	51.54 (↓)	48.10 (↓)
	CLIP-Ref.	1.07	39.12(↓)	49.83 (↓)	44.72(↓)
	TuneCLIP	**9.27**	**65.58** (+2.46)	**69.44** (+0.11)	**63.47** (+1.15)

Experimental Settings

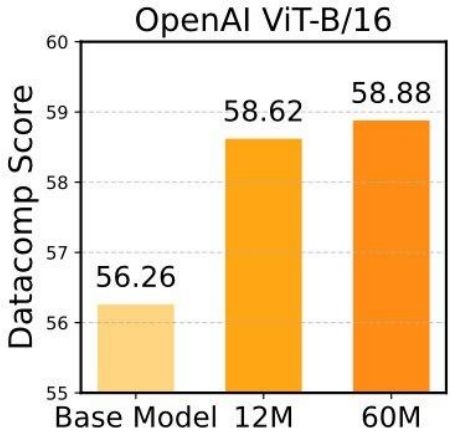
Fine-tuning Data: DFN 12 million and 60 million, CC 12 million

Models: OpenAI CLIPs, LAION CLIP, Google SigLIP, DFN CLIPs

More Insights

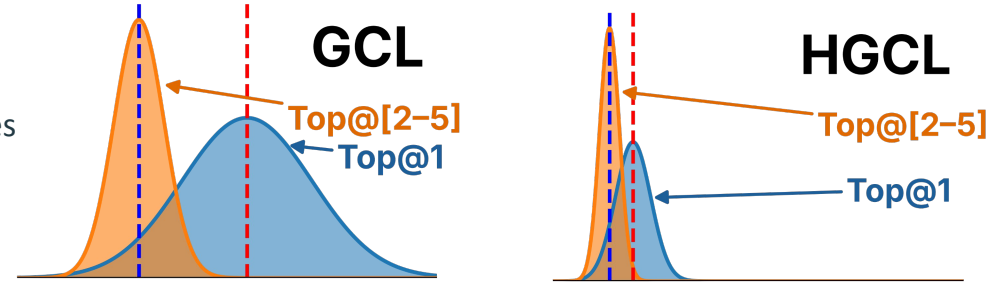


HGCL vs other methods



Data Scaling

Less Penalization of false-negatives



Thank You!