Google DeepMind
EBERHARD KARLS UNIVERSITÄT TÜBINGEN
Tübingen AI Center
HELMHOLTZ MUNICH
TUM

Context-Aware Multimodal Pretraining

CVPR 2025


Karsten Roth

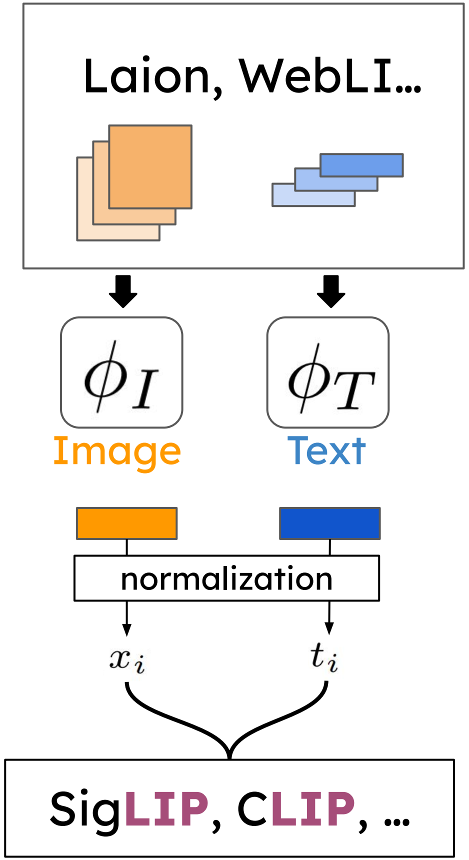

Zeynep Akata


Dima Damen

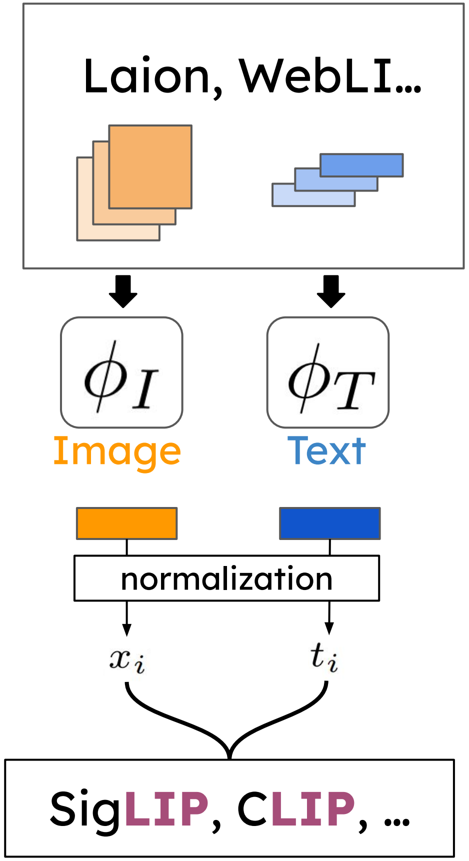

Ivana Balažević


Olivier J. Hénaff

Contrastive Multimodal Pretraining: The vision-encoder workhorse



Contrastive Multimodal Pretraining: The vision-encoder workhorse



SigLIP - Sigmoid-based

$$-\frac{1}{|\mathcal{B}|}\sum_{i,j=1}^{|\mathcal{B}|}\log\frac{1}{1+e^{\mathbb{I}_{i=j}(-\tau_1 x_i t_j + b_1)}}$$

CLIP - Softmax-based

$$\frac{1}{2|\mathcal{B}|}\sum_{i=1}^{\mathcal{B}}\left(\log\frac{e^{\tau_1 x_i t_i}}{\sum_{j=1}^{|\mathcal{B}|} e^{\tau_1 x_i t_h}} + \log\frac{e^{\tau_1 x_i t_i}}{\sum_{j=1}^{|\mathcal{B}|} e^{\tau_1 x_j t_i}}\right)$$

Training in this manner: **High** zero-shot generality, e.g. for open-vocabulary classification, retrieval, …

Going beyond zero-shot generalization?

Modern objectives: Take representations, and re-use them further down the line.
E.g. for retrieval augmentation, memory-augmented models, vision-context in multimodal LLMs…

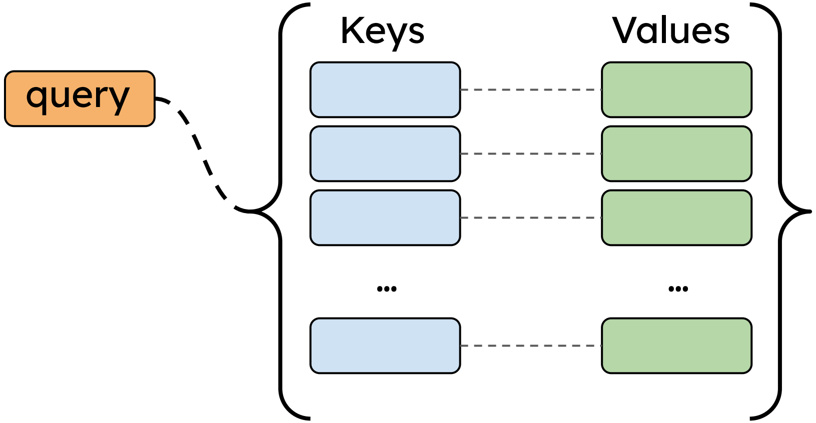
Can you pretrain for such general purpose re-use?

Going beyond zero-shot generalization?

Modern objectives: Take representations, and re-use them further down the line.
E.g. for retrieval augmentation, memory-augmented models, vision-context in multimodal LLMs…

Can you pretrain for such general purpose re-use?

Key perspective:
Representation re-use (e.g. few-shot, many-shot, MLLMs) often involve (soft) dictionary lookup.



Going beyond zero-shot generalization?

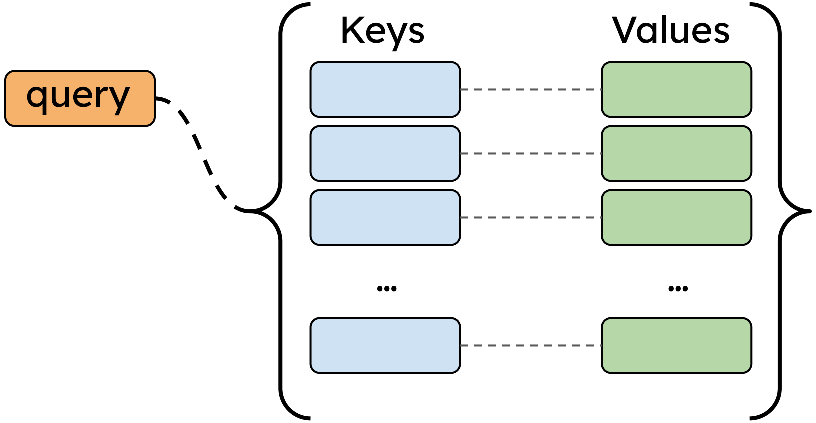
Modern objectives: Take representations, and re-use them further down the line.
E.g. for retrieval augmentation, memory-augmented models, vision-context in multimodal LLMs…

Can you pretrain for such general purpose re-use?

Key perspective:
Representation re-use (e.g. few-shot, many-shot, MLLMs) often involve (soft) dictionary lookup.

Goal:
Pretrain vision encoder at scale to:
- **maintain** zero-shot generality
- be **better** for dictionary lookup style re-use

Important:
Avoid trade-off!



LIxP: Injecting re-usability into pretraining at scale

We introduce Language-Image **Context** Pretraining (LIxP)

Step 1: Take standard objective:

$$\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) = -\frac{1}{|\mathcal{B}|} \sum_{i,j=1}^{|\mathcal{B}|} \log \frac{1}{1 + e^{\mathbb{I}_{i=j}(-\tau_1 x_i t_j + b_1)}}$$

LIxP: Injecting re-usability into pretraining at scale

We introduce Language-Image **Context** Pretraining (LIxP)

Step 1: Take standard objective:

$$\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) = -\frac{1}{|\mathcal{B}|}\sum_{i,j=1}^{|\mathcal{B}|} \log \frac{1}{1 + e^{\mathbb{I}_{i=j}(-\tau_1 x_i t_j + b_1)}}$$

Significant trial & error later...

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\textbf{LIxP}} = \alpha\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\text{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}^{\text{ctx}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha\boxed{\mathcal{L}_{\mathrm{LIP}}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\boxed{\mathcal{L}_{\mathrm{LIP}}}(\mathbf{X}_{\mathcal{B}}^{\mathrm{ctx}}, \mathbf{T}_{\mathcal{B}}, \tau_2)$$

Same loss function

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha\mathcal{L}_{\mathrm{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\mathrm{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}^{\mathrm{ctx}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\textbf{LIxP}} = \alpha \mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\text{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}^{\text{ctx}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

Mimic (soft) dictionary lookup with attention:

$$\boxed{x_i^{\text{ctx}}} = \sigma\left(\frac{\boxed{x_i} \cdot \mathcal{M}_K^T}{\tau_{\text{ctx}}\sqrt{d}}\right)\mathcal{M}_V$$

"Contextualized representation" by looking at memory

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha\mathcal{L}_{\mathrm{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\mathrm{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}^{\mathrm{ctx}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

Mimic (soft) dictionary lookup with attention:

$$\boxed{x_i^{\mathrm{ctx}}} = \sigma\left(\frac{\boxed{x_i} \cdot \boxed{\mathcal{M}_K^T}}{\tau_{\mathrm{ctx}}\sqrt{d}}\right)\boxed{\mathcal{M}_V}$$

What to use as keys and values?

Visual representation, textual, stale, EMA, augmented?

How to efficiently maintain and shard it correctly at scale?

Maintain separate model or large memories can be costly.

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\text{LIP}}(\boxed{\mathbf{X}_{\mathcal{B}}^{\text{ctx}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

Occam's Razor: Simple is best (if done right)

$$\boxed{x_i^{\text{ctx}}} = \sigma\left(\frac{\boxed{x_i} \cdot \boxed{\mathcal{M}_K^T}}{\tau_{\text{ctx}}\sqrt{d}}\right)\boxed{\mathcal{M}_V}$$

$$\hat{\mathbf{X}}_{\mathcal{B}}^{\text{ctx}} = \sigma\left(\frac{\mathbf{M} \odot \mathbf{X}_{\mathcal{B}}\boxed{\mathbf{X}_{\mathcal{B}}^T}}{\tau_{\text{ctx}}\sqrt{d}}\right)\boxed{\hat{\mathbf{X}}_{\mathcal{B}}}$$

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha \mathcal{L}_{\mathrm{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1) + (1-\alpha)\mathcal{L}_{\mathrm{LIP}}(\boxed{\mathbf{X}^{\mathrm{ctx}}_{\mathcal{B}}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

Occam's Razor: Simple is best (if done right)

$$\boxed{x_i^{\mathrm{ctx}}} = \sigma\left(\frac{\boxed{x_i} \cdot \boxed{\mathcal{M}_K^T}}{\tau_{\mathrm{ctx}}\sqrt{d}}\right)\boxed{\mathcal{M}_V}$$

$$\hat{\mathbf{X}}^{\mathrm{ctx}}_{\mathcal{B}} = \sigma\left(\frac{\boxed{\mathbf{M}} \odot \mathbf{X}_{\mathcal{B}}\boxed{\mathbf{X}_{\mathcal{B}}^T}}{\tau_{\mathrm{ctx}}\sqrt{d}}\right)\boxed{\hat{\mathbf{X}}_{\mathcal{B}}}$$

Use same batch as key & values, **but:**

Mask out self-attention shortcut

LIxP: Injecting re-usability into pretraining at scale

Step 2: Introduce a contextualization surrogate objective:

$$\mathcal{L}_{\textbf{LIxP}} = \alpha\mathcal{L}_{\text{LIP}}(\mathbf{X}_\mathcal{B}, \mathbf{T}_\mathcal{B}, \tau_1) + (1-\alpha)\mathcal{L}_{\text{LIP}}(\boxed{\mathbf{X}^{\text{ctx}}_\mathcal{B}}, \mathbf{T}_\mathcal{B}, \boxed{\tau_2})$$

Occam's Razor: Simple is best (if done right)

$$\boxed{x_i^{\text{ctx}}} = \sigma\left(\frac{\boxed{x_i}\cdot\boxed{\mathcal{M}_K^T}}{\tau_{\text{ctx}}\sqrt{d}}\right)\boxed{\mathcal{M}_V}$$

Use same batch as key & values*

$$\hat{\mathbf{X}}^{\text{ctx}}_\mathcal{B} = \sigma\left(\frac{\boxed{\mathbf{M}}\odot\mathbf{X}_\mathcal{B}\boxed{\mathbf{X}_\mathcal{B}^T}}{\tau_{\text{ctx}}\sqrt{d}}\right)\boxed{\hat{\mathbf{X}}_\mathcal{B}}$$

Super scalable extension!

Making it work: A challenge of engineering and patience

- **Three** distinctly learnable, exponential temperatures

- Re-usability is **emergent** - sufficiently long training is needed.

- **Large** enough pretraining batchsizes.

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha \mathcal{L}_{\mathrm{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_1}) + (1-\alpha)\mathcal{L}_{\mathrm{LIP}}(\mathbf{X}^{\mathrm{ctx}}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \boxed{\tau_2})$$

$$\hat{\mathbf{X}}^{\mathrm{ctx}}_{\mathcal{B}} = \sigma\left(\frac{\mathbf{M} \odot \mathbf{X}_{\mathcal{B}}\mathbf{X}_{\mathcal{B}}^T}{\boxed{\tau_{\mathrm{ctx}}}\sqrt{d}}\right)\hat{\mathbf{X}}_{\mathcal{B}}$$

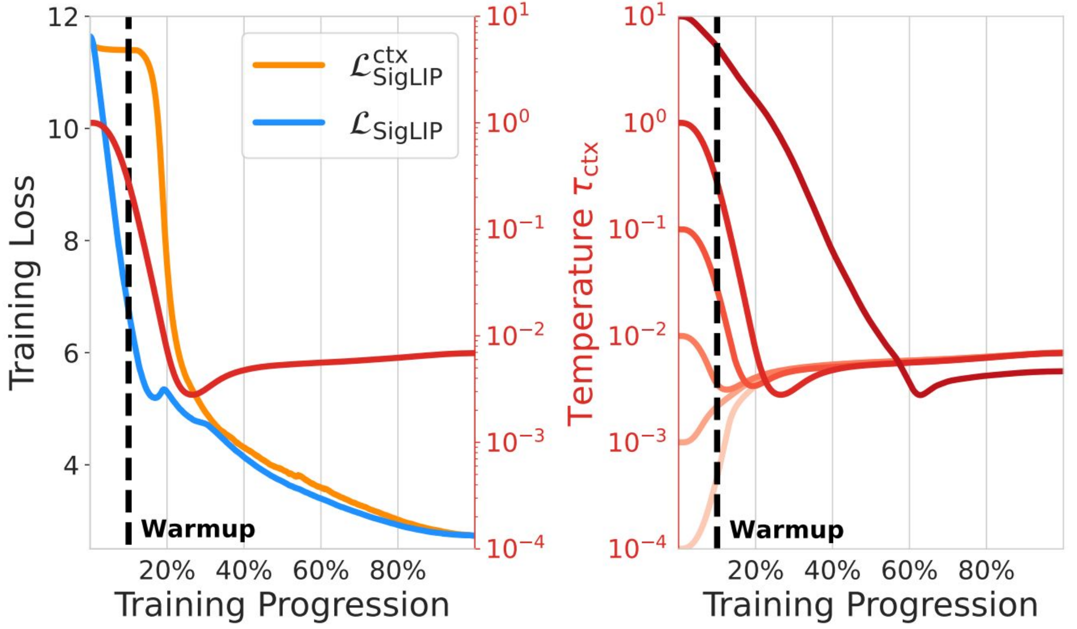
Making it work: A challenge of engineering and patience

Three distinctly learnable, exponential temperatures

Re-usability is **emergent** -
➡ sufficiently long training is needed.

Large enough pretraining batchsizes.

$$\mathcal{L}_{\mathbf{LIxP}} = \alpha \boxed{\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}, \mathbf{T}_{\mathcal{B}}, \tau_1)} + (1-\alpha) \boxed{\mathcal{L}_{\text{LIP}}(\mathbf{X}_{\mathcal{B}}^{\text{ctx}}, \mathbf{T}_{\mathcal{B}}, \tau_2)}$$



Making it work: A challenge of engineering and patience

Three distinctly learnable, exponential temperatures

Re-usability is **emergent** - sufficiently long training is needed.

➡ **Large** enough pretraining batchsizes.

Method	Zero-Shot	16-Shot
SigLIP	51.0	60.1 ± 0.4
Sig**LIxP**	50.5	64.1 ± 0.5
Separate Batch	48.8	63.2 ± 0.3

(a) **Context Batch Separation**

Method	Zero-Shot	16-Shot
None	50.5	64.1 ± 0.5
linear	50.2	62.8 ± 0.2
2-layer MLP	49.8	61.1 ± 0.2
3-layer MLP	49.1	60.8 ± 0.3

(b) **Value Heads for $\mathcal{M}_V$**

Method	Zero-Shot	16-Shot
SigLIP	51.0	60.1 ± 0.4
Sig**LIxP**	50.5	64.1 ± 0.5
No Normalization	48.9	61.7 ± 0.3

(c) **QK Normalization**

Method	Zero-Shot	16-Shot
None	50.5	64.1 ± 0.5
LayerNorm $\{K\}$	46.8	57.1 ± 0.4
LayerNorm $\{V\}$	48.5	62.3 ± 0.5
LayerNorm $\{K, V\}$	48.3	58.2 ± 0.3

(d) **Layer Normalization on $\mathcal{M}$**

Method	Zero-Shot	16-Shot
None	50.5	64.1 ± 0.5
+ Stale (32K)	45.9	59.7 ± 0.3
+ Stale (128K)	46.9	60.5 ± 0.5
+ Stale (512K)	47.2	60.7 ± 0.4

(e) **Inclusion of Stale Buffer**

Method	Zero-Shot	16-Shot
Full (32k)	50.5	64.1 ± 0.5
Subset (1k)	50.7	59.9 ± 0.5
Subset (2k)	50.5	62.9 ± 0.4
Subset (8k)	50.3	63.9 ± 0.3

(f) **Reduced Active Buffer Size**

Method	Zero-Shot	16-Shot
SigLIP	51.0	60.1 ± 0.4
Sig**LIxP**	50.5	64.1 ± 0.5
No masking	50.9	60.5 ± 0.4

(a) **Self-Attention Masking**

Method	Zero-Shot	16-Shot
$\alpha = 0.95$	50.8	62.5 ± 0.3
$\alpha = 0.9$	50.5	64.1 ± 0.5
$\alpha = 0.8$	50.0	63.8 ± 0.3
$\alpha = 0.6$	48.7	61.5 ± 0.4

(b) **Relative weighting**

Method	Zero-Shot	16-Shot
Single-Stage	50.5	64.1 ± 0.5
Residual	49.2	59.2 ± 0.4
Multimodal	47.9	61.4 ± 0.4
Two-Stage	50.5	62.0 ± 0.3

(c) **Contextualization Type**

Method	Zero-Shot	16-Shot
$\tau_1, \tau_2, \tau_{\text{ctx}}$ learnable	50.5	64.1 ± 0.5
$\tau_1 = \tau_2, \tau_{\text{ctx}}$	47.8	61.8 ± 0.4
$\tau_1, \tau_2 = \tau_{\text{ctx}}$	50.4	60.2 ± 0.6
τ_{ctx} frozen	47.1	59.4 ± 0.5

(d) **Uncoupled, learnable temperatures**

Method	Zero-Shot	16-Shot
Full back-propagation	50.5	64.1 ± 0.5
Stop Gradient: $\{K\}$	49.6	62.7 ± 0.4
Stop Gradient: $\{V\}$	43.8	58.7 ± 0.2
Stop Gradient: $\{K, V\}$	44.4	56.5 ± 0.4

(e) **Back-propagation through memory buffer**

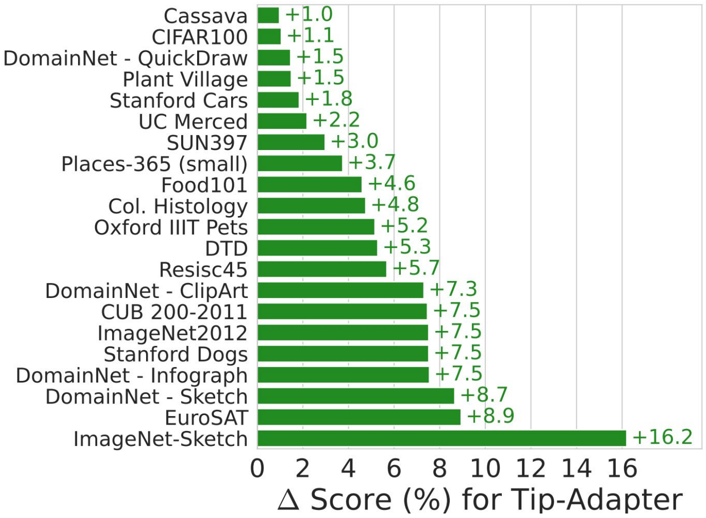
When it works: Near free lunch!

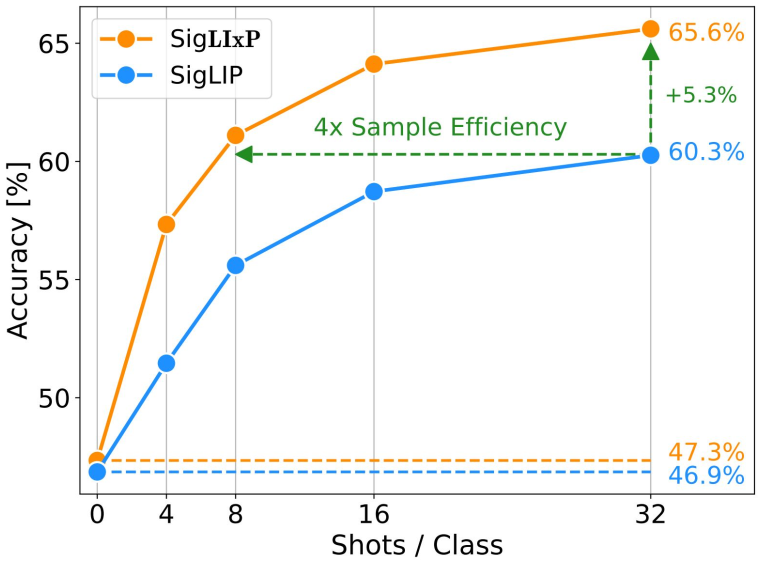
Zero-Shot changes minimal

Model → **Examples** →	ViT-S/16 1.5B	→ 6B	ViT-B/16 6B	→ 15B	ViT-L/16 8B
ZeroShot	46.9 +0.4	52.1 −0.2	60.3 −0.4	62.5 −0.5	64.1 −0.1
Prototypical	57.2 ± 0.2 +4.1	60.8 ± 0.3 +3.4	66.8 ± 0.2 +3.4	67.4 ± 0.3 +3.9	70.7 ± 0.3 +3.3
Default Tip	60.3 ± 0.1 +5.4	63.6 ± 0.3 +4.8	69.5 ± 0.2 +4.3	70.2 ± 0.3 +4.3	73.2 ± 0.3 +4.0
XVal Tip	64.7 ± 0.2 +2.4	67.8 ± 0.3 +2.3	73.8 ± 0.2 +1.6	74.7 ± 0.2 +1.6	77.0 ± 0.3 +1.4
Plurality NN	60.4 ± 0.1 +2.5	63.8 ± 0.2 +1.9	69.1 ± 0.2 +2.3	69.6 ± 0.3 +2.6	72.6 ± 0.2 +1.8
Rank NN	64.8 ± 0.1 +1.7	68.1 ± 0.1 +1.3	73.2 ± 0.1 +1.8	74.1 ± 0.2 +1.8	76.5 ± 0.2 +1.1
Softmax NN	64.2 ± 0.1 +2.8	67.5 ± 0.2 +2.4	72.6 ± 0.2 +2.6	73.3 ± 0.3 +2.7	75.9 ± 0.2 +1.8
Average Gain	**+3.2**	**+2.7**	**+2.7**	**+2.8**	**+2.2**

Few-Shot gains significant

When it works: Near free lunch!

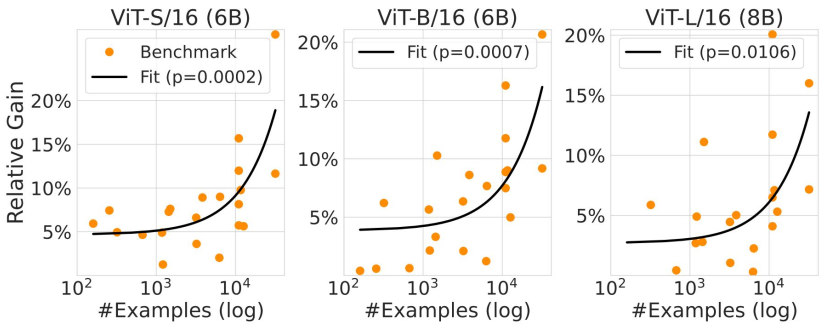




When it works: Near free lunch!

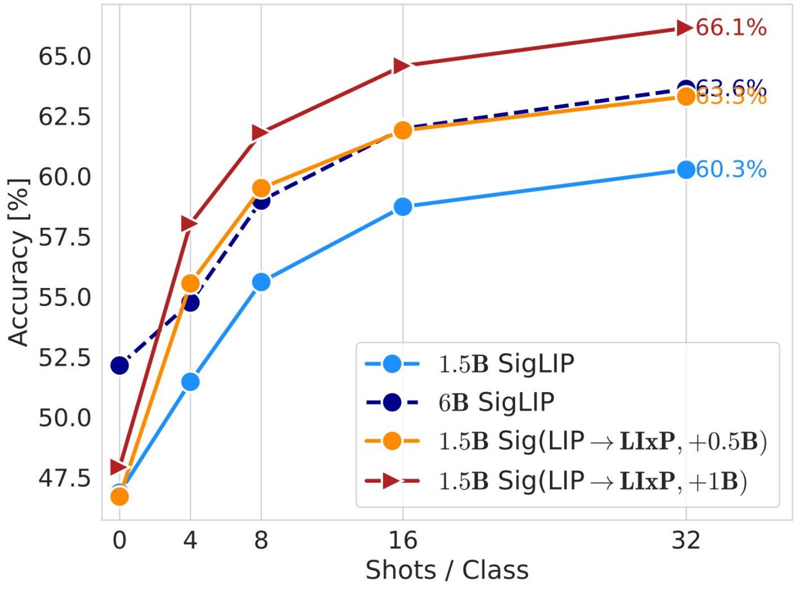
Method	Train-free	IN-1K	DTD	Food101	Pets	Cars
Linear Probe [92]	✗	67.3	70.0	82.9	85.3	80.4
TIP-X [84]	✓	71.1	-	-	-	-
APE [108]	✓	72.1	-	-	-	-
DMN-TF [102]	✓	72.6	71.9	86.0	92.9	78.4
Clip-Adapter [21]	✗	71.1	-	-	-	
MaPLe [36]	✗	72.3	71.3	85.3	92.8	83.6
PromptSRC [37]	✗	73.2	72.7	87.5	93.7	83.8
Tip-Adapter-F [101]	✗	73.7	-	-	-	-
APE-T [108]	✗	74.3	-	-	-	-
CasPL [92]	✗	74.2	75.1	88.4	94.1	86.7
DMN [102]	✗	74.7	75.0	87.1	94.1	85.3
SigLIxP	✓	**77.9**	**76.7**	**92.6**	**94.4**	**92.8**

LIxP pretraining + simple NN classifier
beats
more complex / specialized / learned



The larger your context window / dictionary, the higher the gains!

When it works: Near free lunch!



If **done right**:
can be inject as post-training-stage!

Put together:

It is possible to pretrain for two key functionalities at once, if:

- **Loss balancing**
- **Scalability** via D.o.F in representation space.
- **Dictionary lookup** objective surrogate.
- **Needs scale:** Batchsize informs memory size.
- **Can be post-trained** into a pretrained model too!

Profit!