Appendix

Three Towers: Flexible Contrastive Learning with Pretrained Image Models

A Additional Experiments & Results

A.1 Training Dynamics

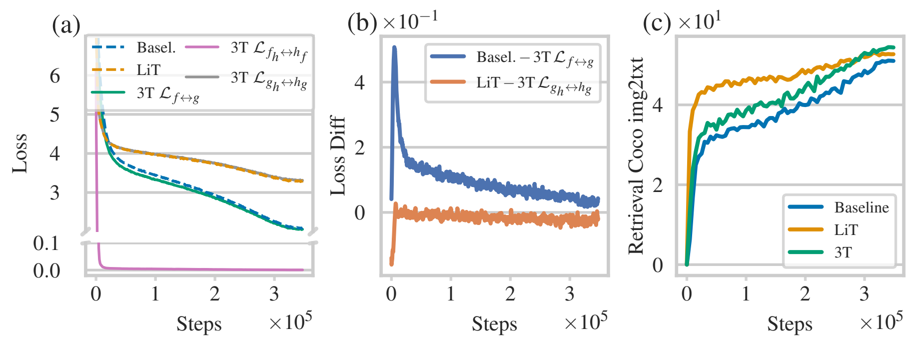

Figure A.1: Training dynamics: (a) The transfer losses, $\mathcal{L}_{f_h\leftrightarrow h_f}$ and $\mathcal{L}_{g_h\leftrightarrow h_g}$, improve the image-text loss, $\mathcal{L}_{f\leftrightarrow g}$, in 3T relative to the baseline. (b) Difference between matching loss terms for 3T, LiT, and the baseline. 3T obtains better image-to-text loss than the baseline and similar locked-image-to-text loss as LiT. (c) While the loss advantage of 3T over the baseline shrinks during training, this does not happen for downstream applications; we display image-to-text retrieval on COCO as an example. Moving averages applied to (a-b) for legibility.

In Fig. A.1, we compare the 3T training losses to LiT and the from-scratch baseline, using the familiar L scale setup with JFT pretraining. For 3T, we display all loss terms separately: the image-text loss $\mathcal{L}_{f\leftrightarrow g}$, the image-to-third-tower loss $\mathcal{L}_{f_h\leftrightarrow h_f}$, and the text-to-third-tower loss $\mathcal{L}_{g_h\leftrightarrow h_g}$, cf. Eq. (4) and Fig. 2. For LiT and the baseline, there is only the image-to-text loss as per Eq. (1). As we train for less than one epoch, we do not observe any overfitting, in the sense that contrastive losses are identical on the training and validation set.

The image-to-third-tower loss, $\mathcal{L}_{f_h\leftrightarrow h_f}$, quickly reduces to near zero, indicating successful knowledge transfer of the pretrained model into the image tower for 3T. Further, $\mathcal{L}_{f\leftrightarrow g}$ behaves similar to the baseline loss; this makes sense because both objectives compute a loss between an unlocked image tower and a text tower. Lastly, $\mathcal{L}_{g_h\leftrightarrow h_g}$ closely follows LiT's loss; this also makes sense because both are losses between a locked pretrained image model and a text tower trained from scratch.

In Fig. A.1 (b), we compute the difference of the baseline (LiT) loss and matching 3T $\mathcal{L}_{f_h\leftrightarrow h_f}$ ($\mathcal{L}_{g_h\leftrightarrow h_g}$) loss, and observe that 3T generally achieves lower (similar) values for the same objective. This suggests a mutually beneficial effect for the individual loss terms of the 3T objective. By aligning the main image and text towers to the pretrained model, 3T obtains improved alignment between the main towers themselves. For the training loss, this effect is large early in training and then decreases. However, for downstream task application, we find that the gap between 3T and the baseline persists; we display retrieval on COCO as an example in Fig. A.1 (c).

A.2 Robustness Metrics

In this section, we study 3T, LiT, and the from-scratch baseline from a robustness perspective, evaluating on a subset of the tasks considered by Tran et al. [71]. Following §4.3, we evaluate all methods across multiple model scales and for both JFT and IN-21k pretraining. We use the full Unfiltered WebLI for all results here. We apply models in zero-shot fashion to these datasets, following the same protocol as for the main zero-shot classification experiments. We continue to use the global temperature τ, cf. §3, learned during training to temper the probabilistic zero-shot predictions.

A.2.1 Probabilistic Prediction and Calibration on CIFAR and ImageNet Variants

In Fig. A.2, we report accuracy, negative log likelihood (NLL), Brier score [22], and expected calibration error (ECE) [47] for 3T, LiT, and the baseline across scales for the following datasets: CIFAR-10, CIFAR-10-C, ImageNet-1k (IN-1k), IN-A, IN-v2, IN-C, and IN-R.

Accuracy. Across all datasets, we find the familiar scaling behavior discussed in §4.2: 3T is consistently better than the baseline, 3T benefits more from increases in scale than LiT, LiT performs well with JFT pretraining but shows weaknesses when pretrained on IN-21k. Note that, for the ImageNet variants, we have previously reported the accuracies (if only at L scale) in §4.2. (Note further, that there might be small discrepancies, because we actually recompute all numbers from a different codebase for the robustness evaluations [15].) For CIFAR-10 [40] and CIFAR-10-C [28], which we have not previously discussed, we also find the familiar scaling behavior. The absolute reduction of performance between CIFAR-10 and CIFAR-10-C is similar across methods, indicating that no approach is significantly more robust to shifts. We observe the same comparing IN-1k to IN-C.

Probabilistic Prediction and Calibration. NLL and Brier scores follow the general trend laid out by the accuracy results. Evidently, the *probabilistic* zero-shot predictions of the methods are all of similar high quality, cf., for example, Tran et al. [71], who investigate probabilistic few-shot predictions. This is confirmed by the ECE results: across tasks, ECE values do not exceed 0.1 at L scale. For 3T, calibration results are regularly better than for LiT, particularly if pretrained on IN-21k, and comparable to those of the baseline: 3T and the baseline have lower calibration error than LiT on 6 out of 7 tasks at L scale with IN-21k pretraining.

We find the low magnitude of the calibration errors surprising. It is striking that the softmax temperature learned during contrastive training would work so well across the various downstream task applications. After all, finding matches across a batch from the contrastive learning dataset and assigning images to labels are, at least superficially, quite distinct tasks. We stress again that no task adaptation of either the models, prompt templates, or softmax-temperatures was performed. We refer to Minderer et al. [45] for a general categorization of our calibration results and discussion in the context of deep learning models.

A.2.2 Out-Of-Distribution Detection

We evaluate the performance of 3T, LiT, and the baseline for out-of-distribution (OOD) detection. We follow the common practice of thresholding the maximum softmax probabilities (MSP) of the models to obtain a binary classifier into in- and out-of-distribution [30, 20]. We report the following metrics: area under the precision-recall curve (AUC(PR)), the area under the receiver operating curve (AUROC), as well as the false positive rate at $95\,\%$ true positives (FPR95). Following Tran et al. [71], we study CIFAR-10 as in-distribution against CIFAR-100, DTD, Places365, and SVHN as out-of-distribution. We also report numbers for IN-1k (in-distribution) vs. Places365 (out-of-distribution).

Typically for OOD evaluations, the model is *trained* on the in-distribution data. Here, we apply methods in a zero-shot manner: we only condition the text tower on the label set of the particular in-distribution dataset. Our image and text towers are trained on the contrastive learning data (image tower trained on JFT/IN-21k for LiT) and *not* adapted to the in-distribution samples. Our contrastive learning methods 'learn' about the in-distribution data only through the label set, and they have to classify each incoming sample as 'in-distribution' or 'out-of-distribution' based solely on how well it aligns with the given set of labels. If a given sample does not match any of the in-distribution labels well, prediction confidence is low, and the sample is classified as OOD. This setup diverges from typical assumptions about OOD experiments and should be interpreted with care. For example, if there were label overlap between the in- and out-of-distribution data (e.g. as would be the case for SVHN vs. MNIST), it would be impossible for the model to classify between in-distribution and OOD without further assumptions. OOD for CLIP/ALIGN-style models has been studied in similar settings by Fort et al. [20], Esmaeilpour et al. [18].

We display results in Fig. A.3. Generally, OOD detection works well with the contrastively learned models, despite conditioning only on the label set: for example, the AUROC for CIFAR10 exceeds 0.95 for both 3T and LiT at L scale for both IN-21k and JFT pretraining. The different metrics, AUC(PR), AUROC, and FPR95, are generally consistent in their ranking across scales and methods. We again find the familiar pattern: 3T is consistently improving over the baseline, and 3T catches up

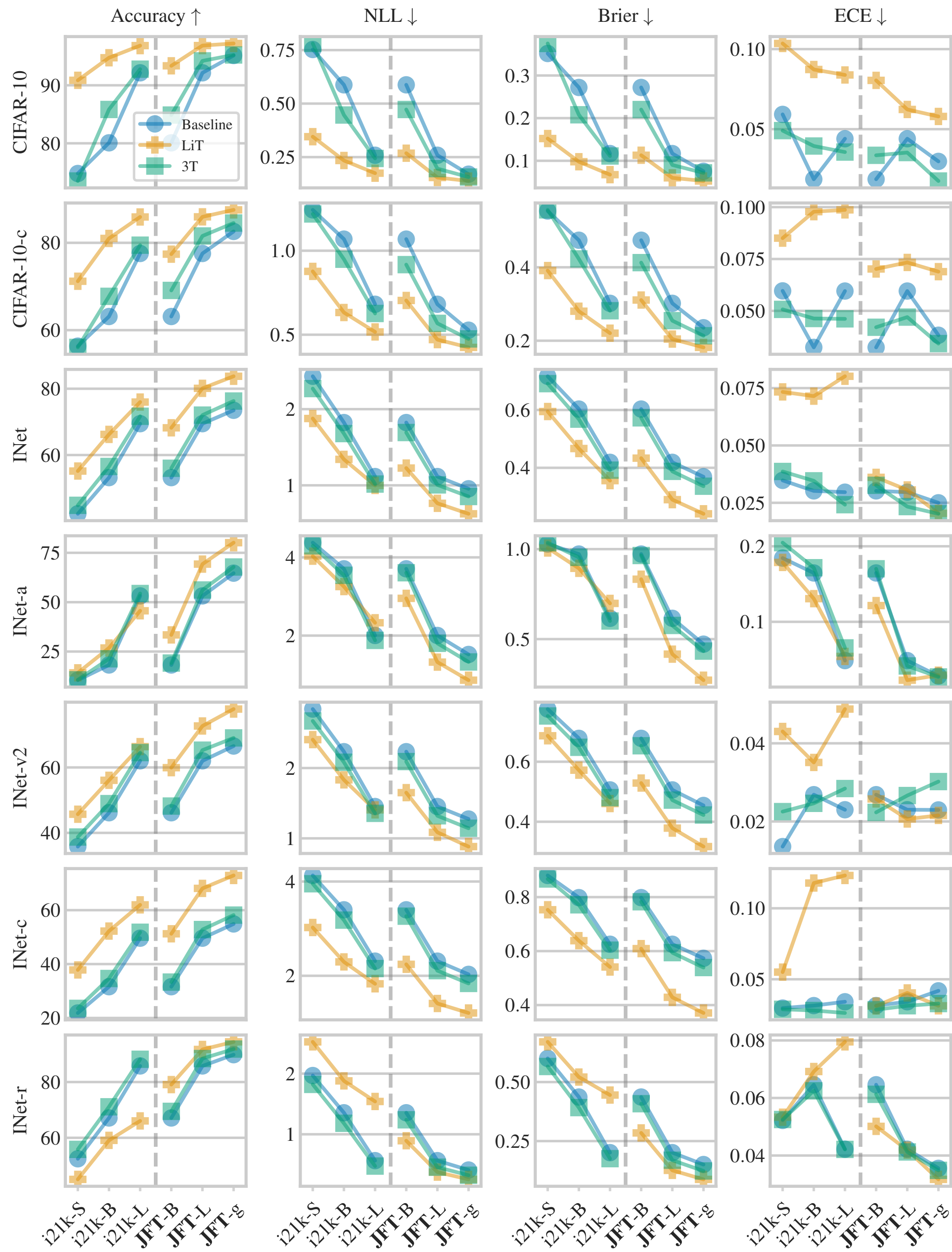

Figure A.2: Robustness evaluation: Accuracy, negative log likelihood (NLL), Brier score, and expected calibration error (ECE) for 3T, LiT, and the baseline for IN-21k and JFT pretraining across scales.

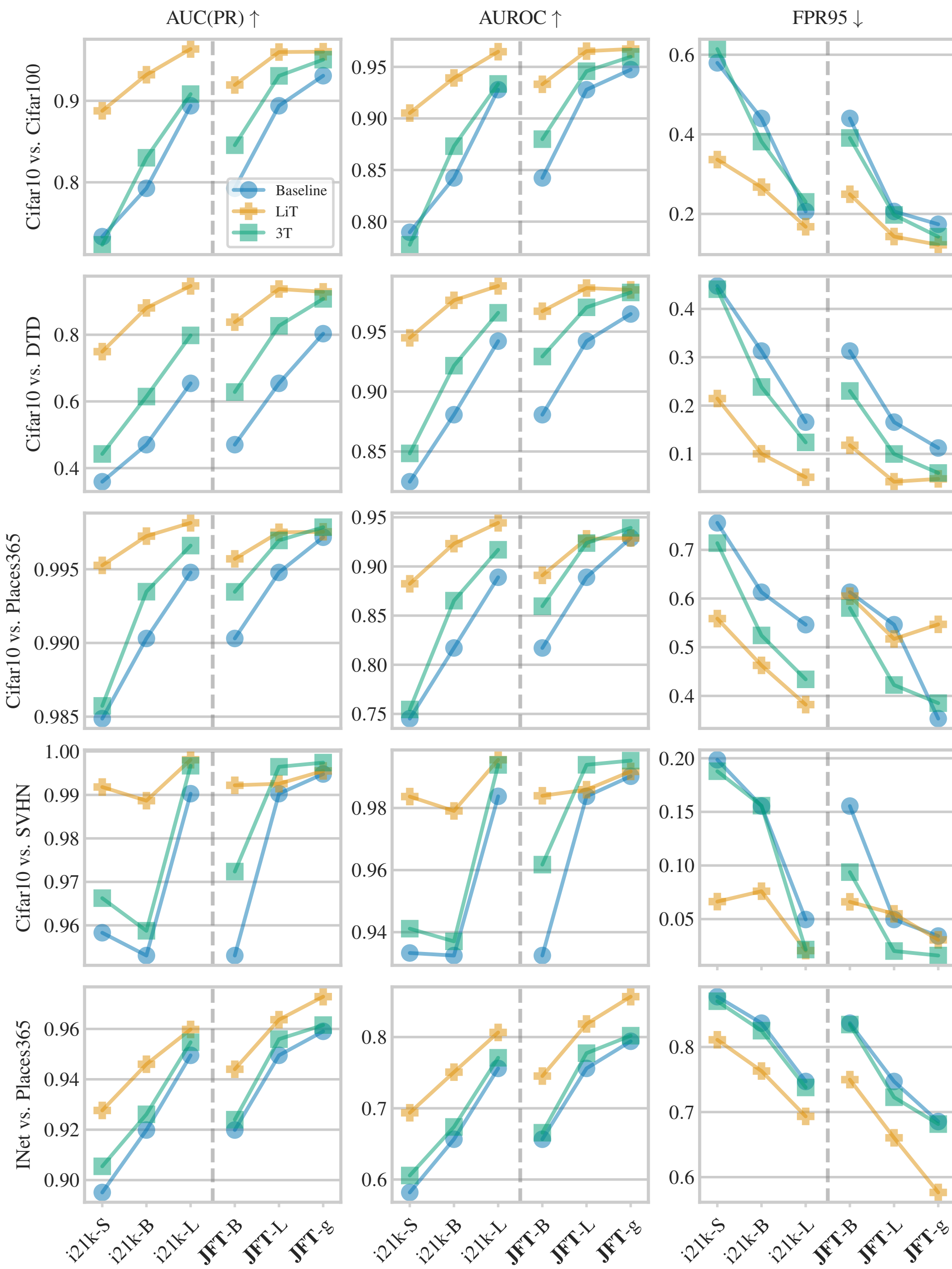

Figure A.3: Robustness evaluation: 3T, LiT, and the baseline for zero-shot out-of-distribution detection (OOD). Reported metrics are area under the precision-recall curve (AUC(PR)), the area under the receiver operating curve (AUROC), and the false positive rate at 95 % true positives (FPR95).

to LiT as scale is increased. For OOD detection, LiT generally does better than 3T and the baseline, perhaps owing to the fact that our choice of CIFAR/IN-1k as in-distribution datasets is advantageous for LiT (similar to how LiT performs particularly well for these datasets for classification).

We find differences between JFT and IN-21k pretraining to be much smaller for the OOD detection task. In fact, in some cases, IN-21k pretraining outperforms JFT pretraining, for example with LiT for the CIFAR-10 vs. Places365 detection task. (This might again be due to the fact that IN-21k pretraining is sufficient for application to CIFAR-10, and only struggles to perform well for other, more varied datasets.) Further, we can observe a rare victory of 3T over JFT-LiT and the baseline at L and g scale in terms of FPR95 on CIFAR-10 vs. Places365 and CIFAR-10 vs. SVHN. Lastly, we see LiT has almost fixed performance at ≈ 0.98 for the CIFAR-10 vs. SVHN task across scales, perhaps due to early task saturation.

A.2.3 JFT Pretraining – Additional Results

In Table A.1, we report few- and zero-shot classification performance for 3T, LiT, and the baseline across our selection of datasets for L scale models and JFT pretraining. LiT outperforms 3T and the baseline on average for few- and zero-shot classification tasks.

In Table A.2, we report performance for g scale models and JFT pretraining across all three splits of the WebLI dataset described in §4. Retrieval performance is generally best for all methods for the Text-Filtered WebLI split, with 3T generally performing best across splits and tasks. For classification, for 3T and the baseline, performance on Text- and Pair-Filtered WebLI is significantly better than on Unfiltered WebLI, with LiT generally performing best across splits. In line with our previous observations, the differences between the WebLI splits are smaller for LiT. As the image tower is kept fixed during contrastive training, LiT performance is influenced less by the contrastive learning setup.

Retrieval Results: Comparison to SOTA. While our retrieval performance is competitive, 3T does not set a new state-of-the art, see, for example, the CoCa paper [80] (Table 3) for a comparison of current methods. While SOTA results were never the aim of this paper—we instead study pretrained models for contrastive learning—there are a few advantages the CoCa setup has, and from which 3T would likely benefit, too. Most notably, CoCa trains for about 6 times more examples seen than we do here (32B vs. 5B). Our scaling experiments, cf. Fig. A.4, suggest we would expect a significant performance increase for longer training. There are further differences that likely benefit CoCa, such as the use of a larger batch size (65k for them vs 14k for us) or training on images with higher resolution for a portion of training (CoCa goes from 288×288 to 576×576, we stay at 288×288)—both of these changes significantly increase computational costs beyond the budget available to us: while CoCa training takes 'about 5 days on 2,048 CloudTPUv4 chips'[80], our g scale runs train for about the same duration on only 512 v4 TPU chips. It would be interesting to see if, in a fairer comparison, 3T matches or outperforms CoCa for retrieval tasks. Alternatively, ideas from 3T could also be used to improve CoCa-like architectures.

Table A.1: For JFT-pretraining, LiT outperforms 3T and the baseline on average on few- and zero-shot classification tasks. L scale models trained on Unfiltered WebLI.

Method		Basel.	LiT	3T
Few-Shot Classification	IN-1k	62.8	81.3	67.7
	CIFAR-100	70.4	83.2	74.3
	Caltech	91.0	89.0	91.8
	Pets	85.9	96.8	88.4
	DTD	70.3	72.1	72.4
	UC Merced	91.8	95.5	93.1
	Cars	81.5	92.9	87.1
	Col-Hist	71.7	81.3	77.0
	Birds	53.4	85.6	62.4
Zero-Shot Classification	IN-1k	69.5	80.1	72.0
	CIFAR-100	73.5	80.1	75.2
	Caltech	81.9	79.5	82.5
	Pets	84.2	96.3	88.7
	DTD	58.6	59.0	59.0
	IN-C	49.6	68.1	52.8
	IN-A	53.0	69.1	56.4
	IN-R	85.8	91.7	88.4
	IN-v2	62.2	74.0	65.4
	ObjectNet	56.2	61.9	59.3
	EuroSat	32.7	36.6	54.7
	Flowers	62.0	76.7	66.6
	RESISC	58.0	58.9	60.9
	Sun397	67.6	69.7	68.1
Average		68.4	77.4	72.4

Table A.2: Results for the baseline, 3T, and LiT for g scale models and JFT pretraining for a selection of different splits of the WebLI dataset. 3T outperforms LiT for retrieval tasks, while LiT performs better for image classification. The from-scratch CLIP/ALIGN-style baseline is not competitive.

Dataset		Unfiltered WebLI			Pair-Filtered WebLI			Text-Filtered WebLI		
Method		Basel.	LiT	3T	Basel.	LiT	3T	Basel.	LiT	3T
Retrieval	Flickr img2txt	75.2	83.0	81.5	81.4	83.2	84.0	85.0	83.9	87.3
	Flickr* img2txt	80.0	84.8	84.2	80.7	83.9	85.6	86.7	85.2	88.3
	Flickr txt2img	58.2	61.3	64.3	61.4	63.9	66.5	67.0	66.5	72.1
	Flickr* txt2img	60.1	63.1	65.6	62.7	65.4	68.4	68.2	67.6	72.9
	COCO img2txt	52.3	57.7	57.5	58.4	59.7	61.7	60.0	59.5	64.1
	COCO txt2img	37.5	40.0	41.1	41.2	41.9	43.9	44.7	43.6	48.5
Few-Shot Classification	IN-1k	67.5	84.6	72.8	71.8	84.6	75.7	69.6	84.6	73.9
	CIFAR-100	72.7	83.2	78.0	73.1	83.2	78.7	76.4	83.2	80.0
	Caltech	91.8	90.0	93.3	89.7	90.0	90.9	90.8	90.0	92.4
	Pets	88.4	97.8	91.5	93.0	97.8	94.3	88.8	97.8	91.4
	DTD	70.7	74.6	74.7	74.2	74.6	76.1	73.6	74.6	76.0
	UC Merced	92.9	96.9	94.7	95.2	96.9	95.6	95.2	96.9	96.5
	Cars	84.1	93.3	88.6	92.6	93.3	93.5	89.0	93.3	91.6
	Col-Hist	72.0	83.6	76.2	77.8	83.6	80.9	73.5	83.6	79.4
	Birds	60.7	89.7	69.8	76.4	89.7	80.7	62.5	89.7	71.1
Zero-Shot Classification	IN-1k	73.5	84.0	76.3	78.0	84.7	79.6	75.8	84.3	78.2
	CIFAR-100	77.5	81.3	80.3	76.2	81.3	79.5	80.6	81.8	82.3
	Caltech	79.8	81.4	82.3	84.0	82.4	82.9	79.5	80.9	81.9
	Pets	87.0	96.4	92.7	92.8	97.7	93.0	88.1	96.5	91.5
	DTD	59.2	62.1	64.9	58.9	55.6	60.1	61.4	62.0	62.1
	IN-C	54.9	72.9	58.2	57.7	73.3	60.3	57.6	73.3	61.3
	IN-A	64.9	80.2	67.8	59.9	79.5	65.1	67.8	80.5	70.8
	IN-R	89.8	94.4	91.8	90.5	94.2	92.8	91.8	94.6	93.3
	IN-v2	66.4	78.1	69.5	70.8	79.2	73.0	69.1	78.5	71.4
	Objectnet	62.7	70.3	65.3	56.9	68.3	59.5	63.3	70.0	65.9
	Eurosat	55.7	33.6	48.9	32.9	30.7	42.8	47.9	36.1	52.1
	Flowers	71.0	84.2	73.5	82.4	86.3	83.0	69.4	86.6	72.5
	RESISC	61.5	58.4	60.5	59.8	56.5	64.8	65.4	57.8	61.7
	Sun397	68.8	71.0	70.3	68.9	71.9	69.8	70.2	71.6	70.9
Average		70.2	77.0	73.7	72.4	77.0	75.3	73.1	77.7	75.9

A.2.4 Pretraining Robustness – Additional Results

In Table A.3, we report results on additional tasks for 3T, LiT, and the baseline for both the 'mismatched' setup and Places365 pretraining. We find again that 3T is much more robust in both setups, significantly outperforming LiT. The difference is particularly striking when using models pretrained on Places365, where LiT's performance degrades drastically while 3T is still able to improve over the baseline.

Table A.3: Testing robustness to the 'mismatched setup' and Places365 pretraining (instead if IN-21k/JFT) for 3T and LiT. In both cases, 3T performs significantly better than LiT. In particular when using models pretrained on Places365, LiT's performance degrades dramatically while 3T continues to improve over the baseline on average. (Note that the baselines here are different not because they use the pretraining dataset, but because we compare to an L scale baseline for the mismatched setup and a B scale baseline (trained for only 900M examples seen) for Places365 pretraining.) We refer to the main text for full details.

Experiment		Mismatched Setup			Places365 Pretraining		
Method		Basel.	LiT	3T	Basel.	LiT	3T
Retrieval	Flickr* img2txt	75.6	66.5	80.2	56.0	35.5	58.1
	Flickr* txt2img	57.1	45.1	62.1	36.2	19.5	38.4
	COCO img2txt	51.0	44.1	54.5	34.1	19.3	36.5
	COCO txt2img	34.2	26.4	37.8	21.0	10.9	22.1
Few-Shot Classification	IN-1k	62.8	70.3	67.6	37.8	16.6	41.5
	CIFAR-100	70.4	80.3	73.8	47.1	33.9	52.7
	Caltech	91.0	88.1	91.7	87.9	66.5	88.5
	Pets	85.9	86.0	86.8	56.8	20.3	59.9
	DTD	70.3	66.3	73.4	58.4	39.7	63.1
	UC Merced	91.8	91.5	93.8	85.8	80.8	89.4
	Cars	81.5	36.7	85.3	57.0	10.1	58.6
	Col-Hist	71.7	84.4	74.3	72.9	70.7	78.7
	Birds	53.4	76.8	65.2	33.2	15.7	38.1
Zero-Shot Classification	IN-1k	69.5	69.5	71.5	45.6	24.5	47.4
	CIFAR-100	73.5	78.6	75.6	48.3	27.4	52.4
	Caltech	81.9	82.0	81.2	76.6	62.7	77.0
	Pets	84.2	84.7	87.4	61.5	30.3	60.2
	DTD	58.6	49.4	60.6	39.8	23.6	39.7
	IN-C	49.6	55.5	51.8	25.3	14.4	27.3
	IN-A	53.0	29.1	54.1	12.0	4.7	12.5
	IN-R	85.8	60.7	87.9	56.1	20.3	58.2
	IN-v2	62.2	61.1	65.0	39.4	20.7	40.5
	Objectnet	56.2	34.9	57.8	28.4	7.3	29.6
	Eurosat	32.7	33.1	52.5	33.7	15.6	27.3
	Flowers	62.0	74.1	66.2	37.6	17.4	37.3
	RESISC	58.0	29.0	57.4	37.9	24.0	38.3
	Sun397	67.6	62.0	68.4	55.1	60.6	57.3
Average		66.4	61.7	69.8	47.5	29.4	49.3

A.3 Scaling Model Sizes and Training Duration – Additional Results

Complementing the results of §4.3, in Fig. A.4 we report the performance when scaling only the number of examples seen during training, keeping the model sizes fixed at B scale. We observe a similar trend to §4.3 / Fig. 3, where 3T benefits more from increases in scale than LiT. Note that, because the dataset size is 10B samples, all of our runs equate to less than a full epoch.

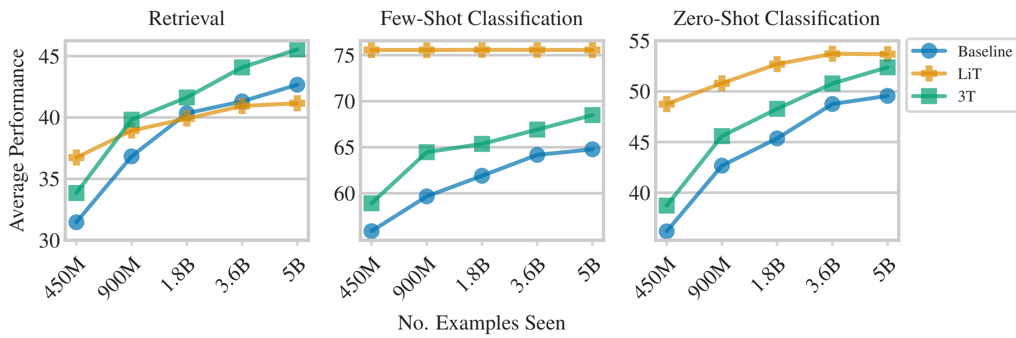

Figure A.4: Increasing training duration of 3T, LiT, and the baseline; average retrieval, few- and zero-shot classification performance. The model scale is B (B/32 for ViTs) for all approaches and towers. 3T and the baseline benefit more from increases in scale than LiT, with 3T maintaining a consistent increase in performance over the baseline. Note that the few-shot performance for LiT is fixed, as only the locked pretrained image tower is used for fewshot applications.

A.4 Benefits From Using 3T With All Three Towers at Test Time – Extended Version

We usually discard the pretrained model when applying 3T to downstream tasks, cf. Fig. 2 (b). Instead, in this section, we explore whether we can find benefits from using the locked third tower at test time, similar to LiT. More specifically, we are interested in interpolating between the main image tower and locked pretrained model in the third tower. Can we interpolate between 3T- and LiT-like prediction by combining the image embeddings?

This idea does not work directly with the default 3T due to our use of linear projection heads, cf. Fig. 2 (a), since there is no unified embedding space that all towers embed to. Therefore, we introduce a 'headless 3T' variant, for which we do not use the linear projection heads, h_f, h_g, f_h, and g_h. (Alternatively, one may think of all linear projection heads fixed to identity mappings.) Thus, all losses directly use the same embeddings, $f(I)$, $p(I)$, and $g(T)$, making the embedding spaces directly comparable. Here, we train B scale models for 3.6B examples seen and use an IN-21k-pretrained model. Further note that the average zero-shot classification performance we report here is over only a subset of the list of tasks used in §4.2: we consider IN-1k, CIFAR-100, and Pets. The selection of few-shot classification and retrieval tasks remains the same, although we do not use the Karpathy split for Flickr here.*

In Fig. A.5, we display the average retrieval, few-shot classification, and zero-shot classification performance for the convex combination, alongside a comparable LiT run and a 3T run with default projection head setup. Across all tasks, we observe similar behavior: for $\alpha = 0$ (full weight on the third tower), we obtain performance close to, but ultimately below, LiT; performance then increases with α, peaking for $\alpha \in [1/4, 3/4]$, before decreasing again. At $\alpha = 1$ (full weight on main image tower), we recover the performance of the headless 3T setup. Interestingly, for retrieval and few-shot classification tasks, the convex combination yields better performance than either of the towers separately across a relatively broad band of α values.

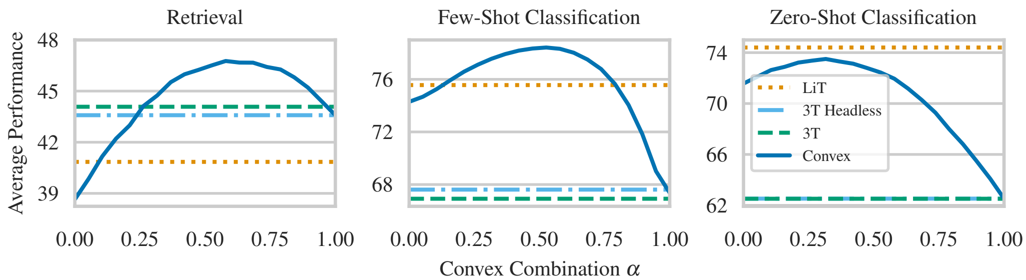

Figure A.5: Convex combination of the image models in 3T: $\alpha \cdot h(I) + (1-\alpha) \cdot f(I)$. By varying α, we can generally interpolate between 3T and LiT performance. Interestingly, for a broad range of weights, the retrieval and few-shot classification performance of the combination outperforms 3T and LiT.

Perhaps counterintuitively, for $\alpha = 0$, we do not recover the performance of LiT exactly. The reasons for this differ between tasks: For retrieval and zero-shot applications, while the image tower is identical to that of LiT, the text tower is different as it has been trained with the 3T objective. For few-shot application, the default evaluation procedure of Zhai et al. [84] uses the prelogits of the ViTs underlying f and h as inputs to the few-shot classifier, i.e. not the final embeddings. As the prelogit spaces of f and h are not aligned, here, we need to instead construct the convex combination in embedding space, which does however mean that $\alpha = 0$ does not give performance equivalent to LiT. Lastly, although the 3T run with the default projection heads does not seem to perform better than '3T headless' in this instance, we have seen 'headless' setups underperform in preliminary experiments and would suggest additional experiments before opting for a headless design, see also §A.5.

We believe that further study of this approach is exciting future work: the method is entirely post-hoc and no additional training costs are incurred, although inference costs do increase.

A.5 Ablation

In this section, we give additional results and details for the ablation study presented in §4.6. Table A.4 gives additional results, extending Table 5 in the main paper. In addition to the mean and two standard errors, we also report standard deviations over tasks here. Note that, for zero-shot classification performance, we only have access to a subset of the full list of tasks used in Section 4.2: we consider IN-1k, CIFAR-100, and Pets. The selection of few-shot classification and retrieval tasks remains the same, although we do not use the Karpathy split for Flickr here.*

No $\mathcal{L}_{\cdot\leftrightarrow\cdot}$ – Details. For this ablation we consider leaving out either of the three loss terms. 'No $\mathcal{L}_{f\leftrightarrow g}$': We replace the 3T loss by $\frac{1}{2} \cdot (\mathcal{L}_{f_h\leftrightarrow h_f} + \mathcal{L}_{g_h\leftrightarrow h_g})$. 'No $\mathcal{L}_{f_h\leftrightarrow h_f}$': We replace the 3T loss by $\frac{1}{2} \cdot (\mathcal{L}_{f\leftrightarrow g} + \mathcal{L}_{g_h\leftrightarrow h_g})$. 'No $\mathcal{L}_{g_h\leftrightarrow h_g}$': We replace the 3T loss by $\frac{1}{2} \cdot (\mathcal{L}_{f\leftrightarrow g} + \mathcal{L}_{f_h\leftrightarrow h_f})$. When leaving out either of the three loss terms, average performance suffers significantly. Leaving out the loss between the main two towers (obviously) has the biggest negative effect, as the main embeddings, $f(I)$ and $g(T)$, are not aligned during training.

Head Variants – Details and Additional Results. In the main part of the paper, we have only given results for the best alternative variant for the projection head setup. Here, we describe all variants and report results individually. We refer to Fig. 2 (a) for the projection head notation. 'Heads only on Third Tower': The main tower projection heads f_h and g_h are fixed to identity mappings. 'Heads Only on Main Towers': The third tower projection heads h_f and h_g are fixed to identity mappings. 'No Heads/Headless': This is the setup described in §A.4: all linear projections h_f, h_g, f_h, g_h are fixed to identity mappings. 'Heads Fully Independent': This setup adds linear projection heads before the computation of $\mathcal{L}_{f\leftrightarrow g}$, i.e. we compute $f_g(I) = \texttt{Lin}(f(I))$ and $g_f(T) = \texttt{Lin}(g(T))$, and then compute the loss $\mathcal{L}_{f_g\leftrightarrow g_f}$ (instead of $\mathcal{L}_{f\leftrightarrow g}$). In Table A.4, we give results for all variants that we try; none outperform the base variant significantly, while some underperform.

MLP Embedding – Details. When replacing the linear projection h in the third tower with an MLP, we use the following architecture: $\texttt{MLP}(x) = \texttt{Lin}_2(\texttt{GELU}(\texttt{Lin}_1(x))$, where we use GELU non-linearities [29], $\texttt{Lin}_1$ expands the embedding dimensionality of the input by a factor of 4, and $\texttt{Lin}_2$ maps to the shared embedding dimension D.

3T with Loss Weights – Details and Additional Results. We replace the standard 3T loss with a weighted objective $\frac{1}{3} \cdot (\mathcal{L}_{f\leftrightarrow g} + w \cdot (\mathcal{L}_{f_h\leftrightarrow h_f} + \mathcal{L}_{g_h\leftrightarrow h_g}))$. For the weights w, we sweep over

Table A.4: Extended results for the 3T ablation study. Difference to the 3T reference run for various architecture ablations. We report mean, standard deviation, and two standard errors of the differences over the downstream task selection.

Difference	Mean	Standard Deviation	Two Standard Errors
Rerun	-0.22	0.50	0.25
No $\mathcal{L}_{f\leftrightarrow g}$	-26.63	21.22	10.61
No $\mathcal{L}_{f_h\leftrightarrow h_f}$	-1.19	1.51	0.75
No $\mathcal{L}_{g_h\leftrightarrow h_g}$	-2.77	1.83	0.91
(Head Variants (best))	0.09	0.70	0.35
Heads Only on Third Tower	0.09	0.70	0.35
Heads Only on Main Towers	-0.67	0.66	0.33
Heads Fully Independent	-0.60	0.63	0.32
No Heads/Headless	-0.47	1.04	0.52
MLP Embedding	-0.08	0.69	0.35
More Temperatures	-0.26	0.95	0.48
(Loss weight = 2 (best))	0.17	1.06	0.53
Loss weight 0.1	-2.31	1.33	0.67
Loss weight 0.5	-0.90	0.81	0.41
Loss weight 2	0.17	1.06	0.53
Loss weight 10	-0.56	1.74	0.87
(L2 Transfer (best))	-3.80	2.27	1.13
L2 Transfer w=0.0001	-4.40	1.89	0.94
L2 Transfer w=0.001	-3.80	2.27	1.13
L2 Transfer w=0.05	-4.41	2.24	1.12
L2 Transfer w=0.01	-4.17	1.97	0.99
L2 Transfer w=.1	-3.97	2.06	1.03
L2 Transfer w=.5	-7.12	2.95	1.48
L2 Transfer w=1	-11.38	4.39	2.19
L2 Transfer w=2	-16.09	5.14	2.57
L2 Transfer w=10	-46.80	14.32	7.16
3T Finetuning	1.85	2.53	1.27

$w \in \{0.1, 0.5, 2, 10\}$. All weights except $w = 2$ lead to an average performance decrease. However, the size of the effect for $w = 2$ is small relative to twice the standard error.

L2 Representation Transfer – Details and Additional Results. We investigate the use of squared losses for the representation transfer between the main towers and the third tower instead of relying on the contrastive loss. Concretely, we replace the 3T loss, Eq. (4), with

$$\frac{1}{3}\left\{\mathcal{L}_{f\leftrightarrow g} + w\frac{1}{N}\sum_{i=i}^{N}\left[\|f_h(I_i) - h_f(I_i)\|^2 + \|g_h(T_i) - h_g(I_i)\|^2\right]\right\}. \tag{5}$$

For the weight hyperparameters w, we sweep over a large set of values, $w \in \{0.0001, 0.001, 0.05, 0.01, 0.1, 0.5, 1, 2, 10\}$. L2 representation transfer gives worse results than the contrastive loss for all values of w we try, corroborating the results of Tian et al. [70].

Finetuning – Details and Additional Results. Initializing the main tower in 3T with the same JFT-pretrained model as the third tower boosts performance significantly, increasing average performance from 56.76 to 58.61. A rerun confirmed these results; we obtained an increase from 56.46 to 58.82. Excited by this, we explored the 3T finetuning setup at other scales, and report performance in Table A.5. Note that here, we increase the numbers of examples seen during training from 450M (S scale) to 900M (B scale) to 5B (L scale). We observe that, as we increase the scale of the experiments, the gains from finetuning the main image tower decrease until they are negligible (compared to rerun variance). We therefore have opted to not make finetuning the main tower part of the standard 3T setup, as it (a) complicates the setup and (b) restricts the main tower to be the same model architecture and scale as the third tower.

Table A.5: Finetuning for 3T: Initializing the main tower in 3T with the same pretrained model as the third tower improves performance significantly at smaller but not larger experiment scales.

Pretraining	Scale	Avg. Performance 3T	Avg. Performance 3T Finetuned
JFT	B	56.76	58.61
	L	73.97	74.22
IN-21k	S	44.39	47.61
	B	56.30	58.83
	L	73.63	73.83

'FlexiLiT 1/2' – Details. With the FlexiLiT variants, cf. Table 5 in the main body of the paper, we investigate if there are other, simple ways to improve LiT. For both variants, we create a new 'half-locked' image tower by adding learnable components to the frozen pretrained image model. For FlexiLiT 1, we add a lightweight learnable 4-layer MLP on top of the frozen backbone: $\texttt{FlexiLiT-1}(I) = \texttt{MLP}(\texttt{LiT}(I))$. The MLP has 4 layers, uses GELU-nonlinearities, and an expansion factor of 4. For FlexiLiT 2, we add an additional learnable ViT next to the locked backbone (adding significant cost) and merge representations with an MLP: $\texttt{FlexiLiT-2}(I) = \texttt{MLP}(\texttt{concat}(\texttt{LiT}(I)), \texttt{ViT}(I))$. The additional ViT is B/32, following the main locked image tower. The MLP merging the two representations is an MLP with the same configuration as for FlexiLiT 1.

B Implementation Details

We follow Zhai et al. [84] for optimization and implementation details. We use the open-source vision transformer implementation available from Beyer et al. [4].

Unless otherwise mentioned, we use Transformers of scale L, with a 16×16 patch size for the ViT image towers, i.e. L/16. We train for 5B examples seen at a batch size of $14 \cdot 1024$, i.e. for about $350\,000$ steps. We resize input images to 224×224 resolution, and normalize pixel values to the $[-1, 1]$ range. Note that for experiments with g scale models, we resize images to 288×288 instead. We use a learning rate of 0.001, warming up linearly for $10\,000$ steps, before following a cosine decay schedule. We use the Adafactor optimizer [64] with default $\beta_1 = 0.9$ and $\beta_2 = 0.99$, and we clip gradients if their norm exceeds 1.0. For g scale runs, we set $\beta_2 = 0.95$ by default, which we found to be important to ensure training stability. We use weight decay of 0.001.

We aggregate embeddings across tokens using multihead attention pooling, i.e. an attention block where the query is a single learned latent vector, and the keys and values are the outputs of the vision transformer (cf. `vit.py` in the code base [4]).

For details on how the different model scales and patch sizes relate to transformer width, depth, MLP dimension, the number of heads, or parameter count, we refer to Table 1 in [17] and Table 2 in [83].

Compute Cost. We train our models on v3 and v4 TPUs. For our main experiments at L scale, we use 256 TPU chips per experiment. Our 3T runs converge in about three days, for example, the 3T run with JFT pretraining took 63 hours of training time to converge over 348772 training steps. The baseline converges in 54 hours, and LiT in 35. For our five main experiments at L scale—3T, LiT for JFT and IN-21k pretraining, and a baseline run—the total runtime was about 280 hours, or about 8 TPU–Chip years worth of compute for the L scale experiments of this project. At g scale, we use 512 TPU chips per run, and our 3T runs converge in about 5 days.

Below we mention additional details pertaining to only some of the experiments.

Details on Few-Shot Classification. Following Zhai et al. [84], we use the prelogits of the ViTs instead of the final embeddings as input to the linear few-shot classifier.

Details on Places Experiment. Following Zhai et al. [84], for the Places365 experiment, we use a B/16 ResFormer [69] as the pretrained model.

C Societal Impact

With 3T, we introduce a novel machine learning method for learning joint embeddings of images and text. We train on large datasets of noisy and potentially biased data crawled from the internet. The same general caveats that apply to CLIP/ALIGN and LiT may also apply to 3T. We refer to §7 in Radford et al. [57] for a general discussion of the societal impact these methods may have.

Additionally, we wish to highlight the importance of carefully evaluating these models, testing for specific undesired behavior, before applying them in production. While the zero- and few-shot classification capabilities of these models are generally impressive, it is also important to consider their limitations and not succumb to wishful thinking when it comes to the real-world performance of these models on arbitrary tasks. For example, all of the approaches we study here do not perform well for zero-shot prediction on the structured and specialized tasks contained in VTAB, which include, for example, medical applications. It is therefore particularly important to carefully evaluate the performance of these methods when applied to real-world applications. Lastly, because 3T and LiT rely on two datasets for training, a classification and a contrastive learning dataset, this can complicate investigations into undesired biases in the final model.

D Libraries & Dataset

We rely on the Jax [5], Flax [26], and TensorFlow [1] Python libraries for our implementation. Additionally, we make use of the Big Vision [4] and Robustness Metrics [15] code bases.

For retrieval performance, we evaluate on Microsoft COCO [10] and Flickr30k [55]. For image classification, we evaluate on IN-1k [40, 60], CIFAR-100 [40], Caltech-256 [23], Oxford-IIIT Pet [52], Describable Textures (DTD) [13], UC Merced Land Use [79], Stanford Cars [39], Col-Hist [37], Birds [72], ImageNet variants -C [28], -A [32], -R [31], -v2 [58], ObjectNet [3], EuroSat [27], Oxford Flowers-102 [49], NWPU-RESISC45 [12], and Sun397 [77].

We take the EuroSat, Flowers, RESISC, and Sun397 datasets from the Visual Task Adaptation Benchmark (VTAB) [82]. They are the only VTAB datasets for which at least one method achieved better than trivial performance.