TernaryCLIP: Efficient Multimodal Distillation with Ternary Quantization

Anonymous ACL submission

Abstract

Recent years have witnessed increasing interest in image-text contrastive modeling, exemplified by models like CLIP, which have been widely used for zero-shot classification and image-text retrieval. In this paper, we propose TernaryCLIP, a lightweight computational framework that converts both the vision and text encoders of CLIP into ternary-weight formats. TernaryCLIP incorporates quantization-aware training and ternarization-aware distillation modules from a full-precision CLIP, enabling low-cost and high-efficiency computing. Comprehensive experiments across 41 real-world datasets demonstrate that TernaryCLIP achieves up to a 16× storage reduction, 60% sparsity, and 2.3× inference acceleration while maintaining competitive accuracy on zero-shot image classification and image-text retrieval tasks. Our work highlights the feasibility of extreme quantization for large multimodal models, supporting effective and efficient deployment on resource-constrained devices.

1 Introduction

Large-scale multimodal models, like Contrastive Language-Image Pretraining (CLIP, Radford et al., 2021), have demonstrated exceptional performance in zero-shot classification, image-text retrieval, and cross-modal understanding. However, their impressive capabilities come at the cost of huge computational and memory demands, making them impractical for resource-constrained environments.

Model quantization has emerged as a promising solution for reducing compute costs by representing model weights with lower precision (Jacob et al., 2018; Wu et al., 2016; Banner et al., 2019). Unlike traditional quantization approaches, such as 8-bit or 4-bit quantization, which achieve a certain degree of memory reduction, recent advances in ternary quantization have demonstrated even greater potential (Zhu et al., 2017a; Li et al., 2016). By compressing the weights into three discrete values: $\{-\Delta, 0, +\Delta\}$, ternary quantization reduces memory consumption and enables efficient bitwise operations during inference (Wang et al., 2019; Micikevicius et al., 2018). Previous works like TernaryBERT (Zafrir et al., 2019) and TTQ (Zhu et al., 2017a) have validated ternarization for unimodal tasks, but the extension to multimodal models like CLIP remains largely unexplored.

In this work, we propose TernaryCLIP (referred to as TnCLIP in the following sections), a fully ternarized version of CLIP that compresses both the vision and text encoders. To preserve the critical image-text alignment capabilities of CLIP, we employ Quantization-Aware Training (QAT) (Nagel et al., 2020; Esser et al., 2020) and integrate Knowledge Distillation (KD) (Hinton et al., 2014; Sanh et al., 2019) from a full-precision teacher model. QAT enables adaptation to ternary constraints during training, mitigating the typical accuracy drop observed in post-training quantization (Gong et al., 2019). Meanwhile, distillation aligns the representations of the ternary student with those of the teacher, minimizing the performance gap (Romero et al., 2015).

We introduce the first ternary quantization of both vision and text encoders of CLIP, achieving substantial model compression while maintaining alignment capabilities through a distillation framework. Our approach achieves up to a 16× storage reduction, 3× memory decrease, 60% sparsity and 2.3× inference acceleration, while preserving competitive classification and retrieval performance. Our results demonstrate that ultra-low-bit multimodal models are feasible, closing the gap between state-of-the-art performance and practical deployment (Shen et al., 2020; Zhu et al., 2017b). TnCLIP enables efficient multimodal learning without sacrificing much accuracy, marking an important step toward deploying vision-language models on resource-constrained devices (Liu et al., 2023).

To promote reproducibility and encourage future research, we are publicly releasing the entire TnCLIP codebase, including model checkpoints. This release includes training and inference scripts, evaluation pipelines, and all settings used in our experiments.

2 Preliminaries

CLIP (Radford et al., 2021) is a multimodal model that aligns images and text in a shared embedding space. Trained on large image-text datasets such as LAION-400M (Schuhmann et al., 2021), CLIP enables tasks such as zero-shot classification, retrieval, and transfer learning. It consists of an image encoder f_i such as ViT and a text encoder f_t such as BERT. Minimizing cosine similarity between image-text pairs while maximizing cosine similarity of negative samples. Let $D = (I_i, T_i)_{i=1}^{|D|}$ denotes dataset used for training.

CLIP is trained with an InfoNCE-based contrastive loss (van den Oord et al., 2018), enabling it to learn meaningful image-text representations. The proposed TnCLIP leverages this contrastive loss, which is illustrated in Figure 1 and formulated as $\mathcal{L}_{\text{task}} = (\mathcal{L}_{\text{I}\to\text{T}} + \mathcal{L}_{\text{T}\to\text{I}})/2$ with

$$\begin{cases} \mathcal{L}_{\text{I}\to\text{T}} = \text{CrossEntropy}(\text{logits}, \text{labels}) \ , \\ \mathcal{L}_{\text{T}\to\text{I}} = \text{CrossEntropy}(\text{logits}^{\text{T}}, \text{labels}) \ . \end{cases}$$

Here, we employ $\text{logits} = I_e(T_e)^T \cdot e^\tau$ to represent the scaled pairwise cosine similarities, where τ is a temperature parameter that controls the scaling of the logits, and I_e and T_e separately are the joint multimodal embeddings of image and text in

$$\begin{aligned} I_e &= \ell_2\text{-normalize}(I_f W_i) \in \mathbb{R}^{n \times d_e} \ , \\ T_e &= \ell_2\text{-normalize}(T_f W_t) \in \mathbb{R}^{n \times d_e} \ , \end{aligned}$$

where $I_f = f_i(I)$ and $T_f = f_t(T)$ are the encoded features, W_i and W_t separately are the projection matrices for image and text features, and the ℓ_2-normalization ensures that the embeddings are unit vectors in the joint embedding space. The labels are the indices of the positive pairs in the batch.

There are several KD methods for CLIP, including CLIP-KD (Yang et al., 2023), ComKD-CLIP (Chen et al., 2024), and SiCLIP (Liu et al., 2024). Guided by Occam's razor principle, TnCLIP employs three KD approaches from CLIP-KD: Contrastive Relational Distillation (CRD), Interactive Contrastive Learning (ICL), and Feature Distillation (FD).

The CRD loss is used to distill the contrastive information from the teacher encoder to a student encoder by aligning the image-to-text and text-to-image distributions through KL-divergence

$$\mathcal{L}_{\text{crd}} = \text{KL}(p^S || p^T) + \text{KL}(q^S || q^T) \ ,$$

where p^S and p^T separately are the image-to-text distributions of student and teacher models, and the KL divergence measures the difference between the text-to-image distributions q^S and q^T.

The ICL loss is used to distill the cross-modal interactive information through anchor-contrastive relationships between student and teacher encoders

$$\begin{aligned} \mathcal{L}_{\text{icl}} &= \frac{1}{2}(\mathcal{L}_{\text{I}\to\text{I}} + \mathcal{L}_{\text{T}\to\text{T}}) \ , \\ \mathcal{L}_{\text{I}\to\text{I}} &= \text{CrossEntropy}(\text{logits}_{\text{I}}, \text{labels}_{\text{I}}) \ , \\ \mathcal{L}_{\text{T}\to\text{T}} &= \text{CrossEntropy}(\text{logits}_{\text{T}}, \text{labels}_{\text{T}}) \ , \end{aligned}$$

where $\text{logits}_{\text{I}} = I_e^S(I_e^T)^T \cdot e^\tau$ indicates the scaled pairwise cosine similarities between student image embeddings and teacher image embeddings, $\text{logits}_{\text{T}} = T_e^S(T_e^T)^T \cdot e^\tau$ is the scaled pairwise cosine similarities between the student and teacher text embeddings, and labels_{I} and labels_{T} denote the indices of the identical positions in the batch, as we aim to interactively align each student embedding with its corresponding teacher embedding.

The FD loss is used to distill direct embedding-level information by minimizing the mean squared error between student and teacher embeddings for both visual and textual modalities

$$\mathcal{L}_{\text{fd}} = \frac{1}{n}\sum_{i=1}^{n} ||I_e^S - I_e^T||_2^2 + \frac{1}{n}\sum_{i=1}^{n} ||T_e^S - T_e^T||_2^2 \ ,$$

where $||\cdot||_2^2$ denotes the mean squared error between the student and teacher embeddings.

3 Methodology

We propose TnCLIP, consisting of ternarization-aware training and ternarization-aware distillation, corresponding to subsection 3.1 and subsection 3.2 for ternarizing model with the reduced resource prerequisite and minimal performance degradation during deployment and inference.

3.1 Ternarization-Aware Training

This subsection presents the Ternarization-Aware Training (TAT) module, which converts weights from full-precision to ternary formats. We start the

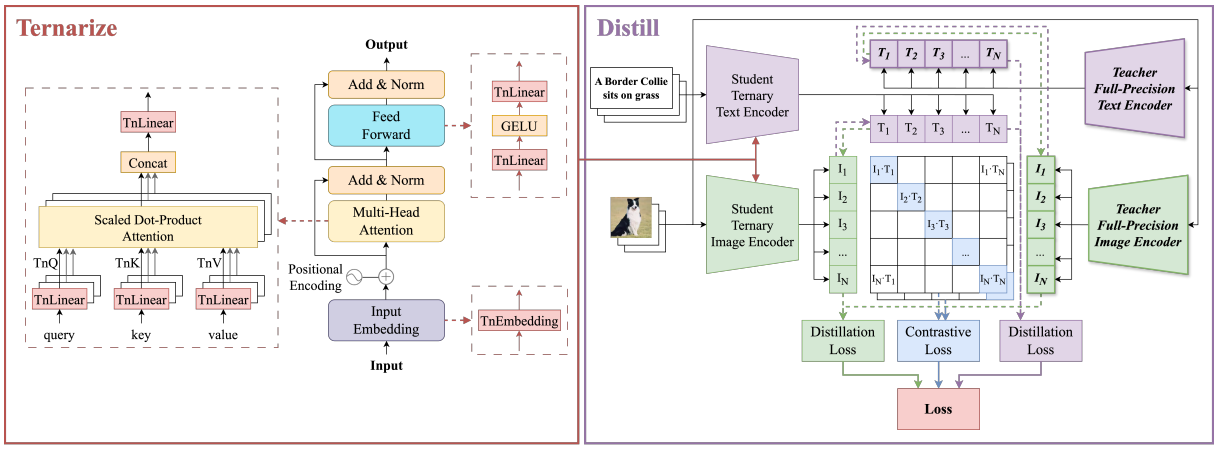

Figure 1: The workflow of the TnCLIP framework, which comprises (a) ternarization-aware training and (b) ternarization-aware distillation.

TAT module with a ternary conversion from the full-precision weights $\mathbf{W}$ to ternary ones $\widetilde{\mathbf{W}}$

$$\widetilde{\mathbf{W}} = \text{RoundClip}\left(\frac{\mathbf{W}}{\gamma+\epsilon}, -1, 1\right) ,$$

where RoundClip clips the rounded values to the range $[a, b]$ according to $\text{RoundClip}(x, a, b) = \max(a, \min(b, \text{round}(x)))$, and the scaling factor γ is computed as

$$\gamma = \frac{\beta}{nm}\sum_{ij}|W_{ij}| ,$$

in which β is the hyperparameter to calibrate ternarization threshold, nm is the total number of elements in the weight matrix, and ϵ is a small constant to prevent division by zero. This scaling factor adaptively adjusts the ternarization threshold based on the magnitude of the weights, preserving better representation of the original weight distribution.

Ternary conversion enables forward propagation with full-precision weights, while gradient updates are applied to the original full-precision parameters. This process is formulated as the following optimization

$$\min_{\mathbf{W}} \quad \hbar(\mathbf{y}, g(\text{ternarize}(\mathbf{W}), \mathbf{x}; \widetilde{\mathbf{W}})) , \qquad (1)$$

where $\hbar$ represents the loss function, $\mathbf{x}$ is the input data, $\mathbf{y}$ denotes the ground truth objective, g is the CLIP model function.

In this work, we solve the optimization in Eq. (1) by backward propagation with gradient descent. Notice that the ternary conversion is inherently non-differentiable, which creates challenges for standard optimizers. Hence, we employ the Straight-Through Estimator (STE) by Bengio (2013) to enable gradient flow through the non-differentiable ternarization operation, i.e., $\partial\mathcal{L}/\partial\widetilde{\mathbf{W}} \leftarrow \partial\mathcal{L}/\partial\mathbf{W}$, where the forward pass uses the ternary weights $\widetilde{\mathbf{W}}$, but gradients flow directly to the full-precision weights $\mathbf{W}$ during backpropagation.

The TAT approach allows the CLIP model to learn parameters that are well-suited for ternary representation while maintaining the optimization stability provided by full-precision updates. Algorithm 1 lists the pseudocode of CLIP equipped with TAT. The TAT algorithm provides a robust framework for adapting CLIP models to efficient ternary representations while preserving model performance. A key strength of this approach is the adaptivity of the weights $\mathbf{W}$, which can originate from the initialization from scratch, the pre-trained weights or the knowledge distilled weights.

3.2 Ternarization-Aware Distillation

TnCLIP framework integrates KD with TAT to develop an efficient ternary CLIP model and ensure that the model learns to perform effectively with ternary representations while benefiting from the knowledge of the high-capacity teacher model. The unified training objective combines both task-specific contrastive learning and teacher-student knowledge transfer, formulated as

$$\mathcal{L}_{\text{ternary}} = \mathcal{L}_{\text{task}}(\mathbf{W}, \mathbf{x}; \widetilde{\mathbf{W}}) + \mathcal{L}_{\text{distill}}(\mathbf{W}, \mathbf{x}; \widetilde{\mathbf{W}}),$$

where the second term indicates the overall KD loss, as shown in Figure 1 and formulated as follows

$$\mathcal{L}_{\text{distill}} = \lambda_{\text{crd}} \cdot \mathcal{L}_{\text{crd}} + \lambda_{\text{icl}} \cdot \mathcal{L}_{\text{icl}} + \lambda_{\text{df}} \cdot \mathcal{L}_{\text{df}} ,$$

where λ_{crd}, λ_{icl}, and λ_{df} are hyperparameters. During training, forward propagation em-

Algorithm 1 Ternarization-Aware Training

Input: Full-precision CLIP model with weights $\mathbf{W}$, dataset $\mathcal{D}$, hyperparameter β
Output: Ternarized CLIP model with weights $\widetilde{\mathbf{W}}$ and scaling factor γ

Initial full-precision weights $\mathbf{W}$
while not converged **do**
  Sample a minibatch (I, T) from $\mathcal{D}$
  // Ternarize weights
  $\gamma \leftarrow \frac{\beta}{nm}\sum_{ij}|W_{ij}|$
  $\widetilde{\mathbf{W}} \leftarrow \text{RoundClip}(\frac{\mathbf{W}}{\gamma+\epsilon}, -1, 1)$
  // Forward pass with ternarized weights
  $I_f, T_f \leftarrow \text{Encoders}(I, T; \widetilde{\mathbf{W}})$
  logits $\leftarrow \text{Similarity}(I_f, T_f; \widetilde{\mathbf{W}}, t)$
  Loss $\leftarrow \text{ContrastiveLoss}(\text{logits})$
  // Backward pass with STE
  $\frac{\partial \text{Loss}}{\partial \mathbf{W}} \leftarrow \text{STE}(\frac{\partial \text{Loss}}{\partial \widetilde{\mathbf{W}}})$
  // Update full-precision weights
  $\mathbf{W} \leftarrow \text{Optimizer}(\mathbf{W}, \frac{\partial \text{Loss}}{\partial \mathbf{W}})$
end while
// Final ternarization
$\gamma \leftarrow \frac{\beta}{nm}\sum_{ij}|W_{ij}|$
$\widetilde{\mathbf{W}} \leftarrow \text{RoundClip}(\frac{\mathbf{W}}{\gamma+\epsilon}, -1, 1)$
return $\widetilde{\mathbf{W}}, \gamma$

ploys ternary weights $\widetilde{\mathbf{W}}$ while maintaining full-precision weights $\mathbf{W}$ for gradient updates. The task loss $\mathcal{L}_{\text{task}}$ preserves the model's ability to align image-text pairs through contrastive learning, while the distillation loss $\mathcal{L}_{\text{distill}}$ transfers complementary knowledge from the teacher model through these three distillation mechanisms: contrastive relational distillation, interactive contrastive learning, and feature distillation.

By simultaneously optimizing for both task and distillation objectives with ternarized weights, the student model learns parameter distributions that are well-suited for ternary quantization while maintaining competitive performance. During inference, the model only operates with the ternary weights, significantly reducing computational and memory requirements without substantial performance degradation.

3.3 Inference Efficiency

TnCLIP uses 1.6875-bit ternary weights[1], achieving a reduction in storage and memory compared to 32-bit representations with the same architecture. This enables deployment on edge devices with limited memory. During inference, matrix multiplication $\mathbf{Wx}$ is optimized with two binary masks $\mathbf{M}_+$ and $\mathbf{M}_-$ representing positive and negative weights

$$\mathbf{Wx} = \Delta \cdot ((\mathbf{M}_+ - \mathbf{M}_-)\mathbf{x}) \,.$$

This replaces floating-point operations with bitwise logic, accelerating computation. Therefore, FLOPs are significantly reduced, as dense matrix multiplications are replaced by bitwise operations. Implementations like FATNN report nearly 2× speedups over 8-bit quantization (Chen et al., 2021). TnCLIP's aggressive weight quantization preserves performance with QAT and KD, making it deployable in memory-constrained settings without sacrificing accuracy.

4 Experiments

We have trained two TnCLIP variants and performed extensive evaluations. The first model is TnCLIP_Q-FFN indicating all feedforward blocks are ternarized, and the second one is TnCLIP_Q-ALL indicating all feedforward and multi-head attention blocks are ternarized. In the following sections, TnCLIP refers to TnCLIP_Q-ALL if there is no specific declaration. TnCLIP applies ternarization on 99% parameters, as shown in Table 1.

	Ternarized		**Ternarized**
	MHA	**FFN**	**Proportion**
TnCLIP_Q-FNN	×	✓	71.62%
TnCLIP_Q-ALL	✓	✓	**99.00%**

Table 1: Comparison of ternarized components and proportion for different model variants.

4.1 Datasets and Configurations

Training Datasets and Models. We use Conceptual 12M (Changpinyo et al., 2021) for vision-and-language ternary distillation with 384 batch size per GPU and 3,072 total batch size. In terms of balancing performance and model size, we select CLIP model with ViT-B/16 image encoder as our student model and LAION's CLIP model with ViT-L/14 image encoder as the teacher model. The more detailed configurations of CLIP models can be checked on Appendex A.

Hyperparameter Tuning. The epoch is set to 32 during hyperparameter tuning and adjust different groups to obtain better loss convergence. Then

[1]Implemented by TQ1_0 in GGML library.

we select the best group of hyperparameters. Basically, we train TnCLIP through an AdamW optimizer with 1e-3 initial learning rate, 10K warm-up steps, cosine annealing scheduler and 0.1 weight decay in 64 epochs. For distillation loss, we set $\lambda_{\text{crd}} = 1.0$, $\lambda_{\text{icl}} = 1.0$, and $\lambda_{fd} = 2000.0$. Check Appendix B for further details on hyperparameters.

Evaluation Tasks and Metrics. The evaluated datasets comprise 37 multi-classification datasets, 1 multi-label classification dataset and 3 image-text retrieval datasets. The tasks are all zero-shot, including standard image classification with the metric of top-1 accuracy (Accuracy@1), multi-label image classification with the metric of mean average precision (mAP), image-text retrieval with the metric of top-5 recall (Recall@5). Considering different CLIP models, CLIP ViT-L/14 LAION is the teacher model of CLIP-KD and TnCLIP variants, while CLIP ViT-B/16 OpenAI and LAION are models for comparison, which are trained from scratch. We benchmark CLIP models (ViT-L/14 and ViT-B/16) and TnCLIP variants (ViT-B/16) on ARM CPU (Apple M4 Pro) to evaluate TnCLIP's sparsity, storage, memory and latency to simulate performance in resource-constrained environments.

Inference Configurations. To eunsure robust and reproducible evaluation on model inference, we employ several configurations. Each benchmark is executed 1,000 times, with average values reported in Table 2 and comprehensive statistics provided in Appendix E. For consistency across experiments, all model weights are stored in the GGUF format[2] and model inference is performed under the GGML framework[3]. For TnCLIP, we employ TQ1_0 quantization data type to implement inference, which efficiently packs and unpacks ternary parameters, optimizing both computational performance and model effectiveness. The benchmarks using alternative quantization data types such as TQ2_0 and TQ4_0 are listed in Appendix E.

4.2 Zero-Shot Classification Performance

Classification Data Types. We assess our TnCLIP and other models on 37 image classification datasets. There are three dataset types: natural, specialized and structured. Natural datasets contain everyday objects and scenes that human commonly encounter, e.g., *ImageNet series* (Deng et al., 2009). Specialized datasets focus on domain-specific application requiring expert knowledge, e.g. *PatchCamelyon(PCam)* covering histopathologic scans of lymph node sections (Veeling et al., 2018). Structured datasets focus on spatial relationships, counting, positioning and other structured reasoning tasks, e.g., *CLEVR*, a synthetic visual question answering dataset (Johnson et al., 2017). The details of all datasets are provided in Appendix D.

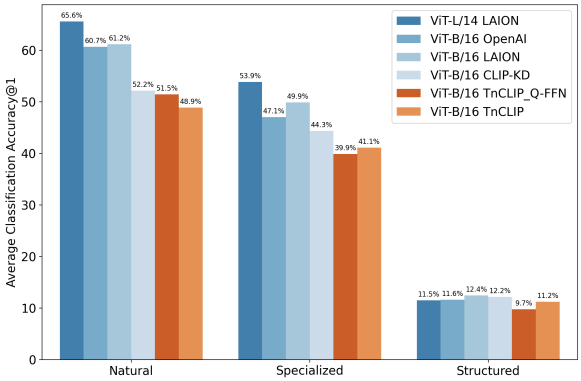

Figure 2: Zero-shot image classification performance (Accuracy@1) on three dataset types and six CLIP models. ViT-L/14 or ViT-B/16 indicates which image encoder is used for the structure of CLIP.

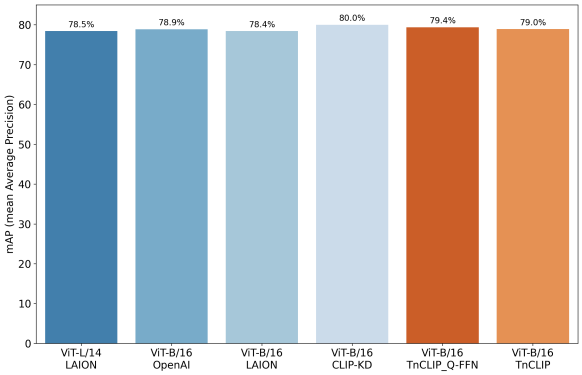

Figure 3: Zero-shot multi-label classification performance (mean average precision) on PASCAL VOC 2007. ViT-L/14 or ViT-B/16 indicates which image encoder is used for the structure of CLIP.

Evaluation of Accuracy@1. Figure 2 illustrates TnCLIP variants can handle performance degradation due to quantization error and the performance gap is less than 3% compared to CLIP-KD, considering all evaluated dataset categories. Compared to the train-from-scratch models, TnCLIPs appear around 8% performance reduction, while compared to the teacher model the reduction is around 11% in Figure 2 and Table 7. Furthermore, all knowledge distilled models (CLIP-KD and TnCLIPs) demonstrate promising performance improvement, around

[2] https://github.com/ggml-org/ggml/blob/master/docs/gguf.md

[3] https://github.com/ggml-org/ggml

20% compared to 36% Accuracy@1 on ImageNet-1K by the CLIP model without distillation (Yang et al., 2024). For the zero-shot multi-label classification task, TnCLIP variants maintain mAP scores within 1% margin of other CLIP models in Figure 3.

Evaluation of Sparsity, Storage, memory and latency. In Table 2, we observe substantial improvements across all evaluation metrics for TnCLIP models. The extremely compressed variant TnCLIP (ViT-B/16) achieves 60.88% total sparsity, 35.3MB storage, 8MB runtime memory footprint and 106.75ms latency. For the 60.88% total model sparsity, TnCLIP provides the possibility for further optimization and compression in subsequent research that other full-precision models do not. Regarding storage requirements, TnCLIP achieves an order-of-magnitude reduction, shrinking the model size from 1630.9MB (ViT-L/14 Float32) and 571.6MB (ViT-B/16 Float32) to merely 35.3MB which represents 94%~98% decrease. Similarly, memory during inference is dramatically optimized with a reduction from 24MB~60MB to 8MB, enabling deployment on memory-constrained devices. With respect to inference latency, TnCLIP delivers remarkable acceleration reducing processing time from 241.40ms~886.15ms to 106.75ms, which achieves a 56%~87% improvement.

Zero-shot classification experiments demonstrate that our proposed TnCLIP retains competitive effectiveness and achieves great efficiency in sparsity, storage, memory and latency.

4.3 Zero-Shot Retrieval Performance

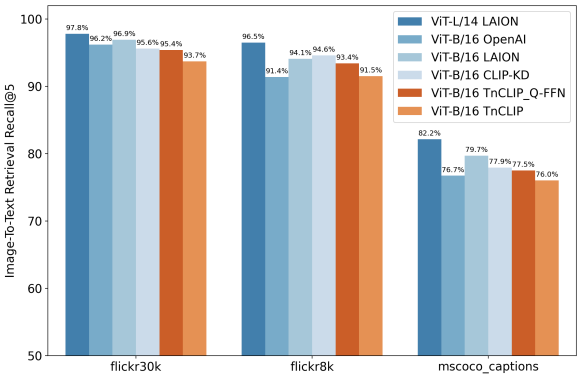

Figure 4: Zero-shot image-to-text retrieval performance (Recall@5) on three datasets and six CLIP models. ViT-L/14 or ViT-B/16 indicates which image encoder is used for the structure of CLIP.

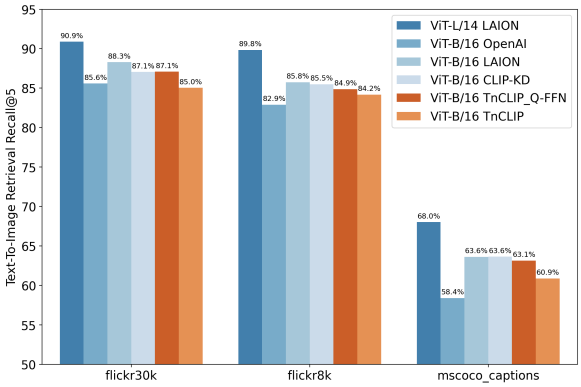

Figure 5: Zero-shot text-to-image retrieval performance (Recall@5) on three datasets and six CLIP models. ViT-L/14 or ViT-B/16 indicates which image encoder is used for the structure of CLIP.

Datasets. For the evaluation in the zero-shot cross-modal retrieval scenario, we apply three commonly used datasets: Flickr8k (Hodosh et al., 2013), Flickr30k (Young et al., 2014) and MS COCO (Microsoft Common Objects in Context, Lin et al. (2014)) on two tasks: image-to-text retrieval and text-to-image retrieval. We choose Recall@5 as the metric to judge the retrieval performance.

Evaluation. In Figure 4 and Figure 5, it is shown that the performance reduction is less than 3% relative to the same model size, while around 5% relative to the larger model size. TnCLIP achieves competitive performance compared to CLIP models with larger architecture and higher precision. In terms of model sparsity, storage, memory and latency, TnCLIP performs much better than full-precision models which is elaborated in the above section 4.2.

4.4 Covergence and Costs

Covergence. As shown in Figure 6, we analyze the loss curves and obtain three crucial findings, while more detailed training information can be seen in Appendix C. Firstly, ternary loss curves exhibit periodic higher fluctuations as the full-precision loss curve does not. It is reasonable since unlike continuous parameter space of the full-precision models, quantized or ternarized parameter space is discrete by $roundclip(x, -1, 1)$ function which resulting in slight weight updates but great performance differences between two training steps. Secondly, scaling up the quantized proportion introduces increased quantization error and more computation to mitigate it. Lastly, more training computation

Model	Precision	Sparsity ↑	Storage(MB) ↓	Memory(MB) ↓	Latency(ms) ↓
CLIP ViT-L/14	Float32	0%	1630.9	60	886.15
	Float16	0%	817.3(0.50×)	30(0.50×)	710.88(0.80×)
CLIP ViT-B/16	Float32	0%	571.6(0.35×)	24(0.40×)	241.40(0.27×)
	Float16	0%	287.7(0.18×)	12(0.20×)	174.62(0.20×)
TnCLIP_Q-FFN ViT-B/16	TQ1_0	44.57%(62.27%)	105.0(0.06×)	8(0.13×)	126.05(0.14×)
TnCLIP ViT-B/16	TQ1_0	**60.88%(61.56%)**	**35.3(0.02×)**	**8(0.13×)**	**106.75(0.12×)**

Table 2: CLIP model sparsity, storage, memory and latency on different precisions. For the sparsity of ternary models, x%(y%): x% represents total sparsity of all parameters and y% represents sparsity of all ternary parameters.

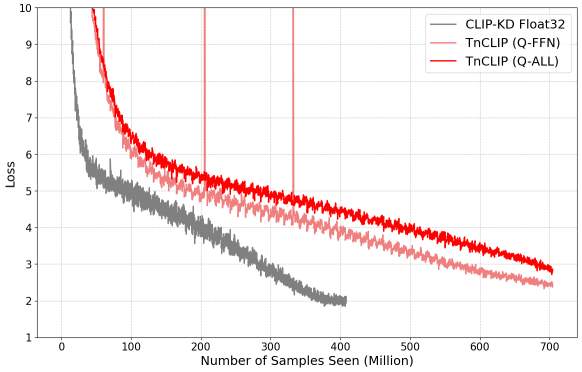

Figure 6: Training losses of three models on the number of samples seen: 1) full-precision CLIP-KD, 2) TnCLIP (Q-FFN) with FFN ternarized, 3) TnCLIP (Q-ALL) with MHA+FFN ternarized.

can mitigate the quantization error (or performance barrier) contributed by the quantization operation to gain a promising performance compared to the full-precision model.

Above all, inspired by the scaling law in large language models (Kaplan et al., 2020), we summarize the scaling law of 1.6875-bit quantization: increased compute can mitigate performance reduction.

$$L_q(C, q) = \alpha C^{\beta} \cdot f(q) + c,$$

where $L_q(C, q)$ is the quantized model loss. C is the training computational budget. α and β are the power law parameters. $f(q) = (1 - q)^{\gamma}$ is the quantization penalty function where q is the quantized proportion and γ is the parameter. c is the regulation.

Computational Costs. Apart from TnCLIP, the models appearing in the evaluation experiments are sourced from OpenAI (Radford et al., 2021), LAION[4], and CLIP-KD (Yang et al., 2024). Although the training data and computational resources used to train a CLIP with ViT-B/16 image encoder by OpenAI are not released, these are announced by LAION and CLIP-KD. In this case, we compare our model TnCLIP with models from OpenAI, LAION and CLIP-KD about the computational usage and the performance. We collect GPU usage information for different models in Table 3 and the prices are calculated from the prices to rent on Vast.ai[5]. It is obvious that training TnCLIP is not only significantly more cost-effective than distilling by CLIP-KD but also training from scratch by OpenAI and LAION.

4.5 Discussions

Comprehensive experiments reveal several key insights into ternary quantization for multimodal models. First, we observe a scaling law for 1.6875-bit quantization, where the performance degradation caused by quantization error can be effectively mitigated by increasing computational resources during training, specifically training iterations and dataset size. This finding aligns with recent work on QAT in large language models (Wang et al., 2023), indicating that the scaling law generalizes across modality domains.

Second, TnCLIP effectively achieves the primary objectives of model ternarization: (1) Reduced training cost through knowledge distillation. (2) Performance preservation with minimal degradation via ternarization-aware training, (3) larger sparsity, along with lower storage, memory and latency through efficient ternary operations.

Lastly, the performance trade-offs observed in some evaluation metrics are acceptable given the resource efficiency gains, especially for resource-constrained environments, e.g., edge devices. Experiments demonstrate that the performance gap between full-precision (CLIP-KD) and ternary mod-

[4] https://laion.ai/

[5] https://vast.ai/pricing

Model	Dataset	GPU	Time ↓	Price ↓
ViT-L/14 LAION	LAION-400M	400* A100	∼127 hours	∼44,704$
ViT-L/14 OpenAI	WIT-400M	256* V100	∼288 hours	∼34,818$
ViT-B/16 LAION	LAION-400M	176* A100	∼61 hours	∼9,448$
ViT-B/16 CLIP-KD	CC3M+CC12M	8* A800	∼137 hours	∼998$
ViT-B/16 TnCLIP	CC12M	8* 6000Ada	∼129 hours	∼**575$**

Table 3: Comparison of CLIP model training specifications, including vision transformer size, dataset, GPU requirements, training time, and associated costs.

els (TnCLIP) under the same model architecture is minimal ($\leqslant$2.7% on average in Table 7), while resource consumption decreases by approximately one order of magnitude in Table 2.

5 Related Work

Model Quantization and Ternarization. Quantization has been a key strategy to compress neural networks for efficient deployment. Traditional methods like integer quantization (INT8 and INT4) effectively reduce memory usage and computation costs with minimal accuracy loss (Jacob et al., 2018; Wu et al., 2016). Recent research extends to ternary quantization, where weights are represented as $\{-1, 0, +1\}$, achieving even greater efficiency while preserving accuracy (Zhu et al., 2017a; Zafrir et al., 2019). Techniques such as TTQ (Zhu et al., 2017a) and FATNN (Chen et al., 2021) introduce learned scaling factors to optimize ternary models. Our work extends ternary quantization to multimodal learning, fully quantizing both the visual and textual encoders of CLIP.

Transformer Quantization. Transformer quantization for vision and language processing has been explored in works such as Q8BERT and TernaryBERT, demonstrating low-bit precision with minimal performance degradation (Zafrir et al., 2019; Zhang et al., 2020). In the vision domain, quantizing ViTs presents challenges due to sensitivity in attention distributions, which methods such as Q-ViT address with quantization-aware adjustments (Li et al., 2022). Building on these ideas, our TnCLIP is the first to apply ternary quantization to CLIP's dual-modality architecture.

Multimodal Model Compression. Compression for multimodal architectures remains underexplored compared to unimodal models like BERT or ResNet. Recent efforts such as TinyCLIP (Wu et al., 2023) employ distillation to reduce model size, focusing on parameter reduction rather than quantization. Although 8-bit quantization has demonstrated memory and latency benefits for CLIP (Jacob et al., 2018; Zafrir et al., 2019), our TnCLIP achieves 1.6875-bit weight compression across both image and text encoders, setting a new benchmark for efficient multimodal learning.

Knowledge Distillation for Low-Bit. KD has been effective in boosting performance in compressed models. Techniques like TinyBERT (Wu et al., 2023) and DistilBERT (Sanh et al., 2019) demonstrate that distillation from high-precision teachers can bridge accuracy gaps in quantized students. Our TnCLIP leverages KD not only for knowledge transfer but also for mitigating representational loss from ternary quantization, ensuring robust cross-modal retrieval.

Multimodal Model Efficiency. As multimodal models continue to scale up, computational efficiency for edge deployments becomes crucial. Although pruning, distillation, and low-bit quantization have been explored for unimodal models (Hinton et al., 2014; Han et al., 2016; Esser et al., 2020), ternary quantization remains underrepresented. Our TnCLIP pioneers the full ternarization of a vision-language transformer, offering substantial memory and compute reductions without sacrificing cross-modal alignment, paving the way for efficient multimodal learning in resource-constrained environments.

6 Conclusions and Prospects

We presented TnCLIP, which compresses both the vision and text encoders of CLIP into ternary formats. By integrating QAT and KD, TnCLIP achieves up to 16× storage reduction, 60% sparsity, and 2.3× inference acceleration while maintaining competitive classification and retrieval accuracy. Experimental results demonstrate its effectiveness, supporting efficient deployment on resource-constrained devices such as mobile phones and AR/VR applications.

7 Limitations

Although TnCLIP is efficient due to ternary quantization, it has several limitations. First, ternatization-aware distillation from pre-trained models reduces the student model's ability to adapt to domain-specific applications. Retraining is still necessary to maintain performance on new tasks. Second, quantization introduces information loss, which can hinder the model's ability to capture subtle image-text alignments compared to the full-precision models trained from scratch.

Additionally, the current TnCLIP framework only quantizes weights, leaving activations in FP16 format. Furthermore, the current implementation does not fully leverage mixed-precision techniques, such as using INT8 for activations and intermediate computations, even though these techniques offer promising trade-offs.

Finally, this work focuses on image-text alignment tasks. The extension of ternary quantization techniques to other modalities remains unexplored and presents an opportunity for future research.

8 Ethical Considerations

TnCLIP inherits the biases of its pre-trained teacher models, which may reflect societal biases in image-text alignment tasks. Although ternary quantization and distillation methods do not introduce new biases, they do not specifically mitigate inherited representation biases. Furthermore, the model's reduced interpretability due to aggressive quantization complicates the detection and mitigation of biases. Future work should explore bias-aware quantization techniques to improve fairness and transparency when deploying the model in real-world applications.

References

Ron Banner, Elad Nahshan, and Daniel Soudry. 2019. Post-training 4-bit quantization of convolutional networks for rapid deployment. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Yoshua Bengio. 2013. Estimating or propagating gradients through stochastic neurons. *arXiv preprint arXiv:1305.2982*.

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. 2021. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Peng Chen, Bohan Zhuang, and Chunhua Shen. 2021. Fatnn: Fast and accurate ternary neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5199–5208. IEEE.

Yifan Chen, Xiaozhen Qiao, Zhe Sun, and Xuelong Li. 2024. Comkd-clip: Comprehensive knowledge distillation for contrastive language-image pre-training model. *arXiv preprint arXiv:2408.04145*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE Computer Society.

Steven K Esser, Jeffrey L McKinstry, Divya Bablani, Raja Appuswamy, and Dharmendra S Modha. 2020. Learned step size quantization. In *International Conference on Learning Representations (ICLR)*.

Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. 2019. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Song Han, Huizi Mao, and William J Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *International Conference on Learning Representations (ICLR)*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2014. Distilling the knowledge in a neural network. In *Neural Information Processing Systems (NeurIPS) Deep Learning Workshop*.

Micah Hodosh, Peter Young, and Julia Hockenmaier. 2013. Framing image description as a ranking task: Data, models and evaluation metrics. In *Journal of Artificial Intelligence Research*, volume 47, pages 853–899.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Benoit Steiner, and Jason Rolfe. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2704–2713. IEEE.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Fengfu Li, Bo Zhang, and Bin Liu. 2016. Ternary weight networks. In *arXiv preprint arXiv:1605.04711*.

Yanjing Li, Sheng Xu, Baochang Zhang, Xianbin Cao, Peng Gao, and Guodong Guo. 2022. Q-vit: Accurate and fully quantized low-bit vision transformer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C Lawrence Zitnick, and Piotr Dollár. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755.

Hongbo Liu and 1 others. 2024. Simplifying clip: Unleashing the power of large-scale models on consumer-level computers. *arXiv preprint arXiv:2411.14789*.

Xinyu Liu, Houwen Peng, Ningxin Zheng, Yuqing Yang, Han Hu, and Yixuan Yuan. 2023. Efficientvit: Memory efficient vision transformer with cascaded group attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and 1 others. 2018. Mixed precision training. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. 2020. Up or down? adaptive rounding for post-training quantization. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pages 7197–7206. PMLR.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139, pages 8748–8763. PMLR.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations (ICLR)*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert: A distilled version of bert: Smaller, faster, cheaper and lighter. In *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (NeurIPS)*.

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. 2021. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*.

Sheng Shen, Zhewei Dong, Xueying Ye, Lin Ma, Zhenyu Wang, and Chuang Gan. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *Preprint*, arXiv:1807.03748.

Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. 2018. Rotation equivariant cnns for digital pathology. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*, pages 210–218. Springer.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*.

Kuan Wang, Zhijian Zhang, Haichuan Liu, Yujun Wei, Bojian Bao, Chuang Gan Yang, and Song Han. 2019. Haq: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8604–8612. IEEE.

Jianxin Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Rongrong Cheng. 2016. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4820–4828. IEEE.

Kan Wu, Houwen Peng, Zhenghong Zhou, Bin Xiao, Mengchen Liu, Lu Yuan, Hong Xuan, Michael Valenzuela, Xi Chen, Xinggang Wang, Hongyang Chao, and Han Hu. 2023. Tinyclip: Clip distillation via affinity mimicking and weight inheritance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chuanguang Yang, Zhulin An, Libo Huang, Junyu Bi, Xinqiang Yu, Han Yang, Boyu Diao, and Yongjun Xu. 2023. Clip-kd: An empirical study of clip model distillation. *arXiv preprint arXiv:2307.12732*.

Chuanguang Yang, Zhulin An, Libo Huang, Junyu Bi, Xinqiang Yu, Han Yang, Boyu Diao, and Yongjun Xu. 2024. Clip-kd: An empirical study of clip model distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78.

Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8-bit bert. In *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (NeurIPS)*.

Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. 2020. Ternarybert: Distillation-aware ultra-low bit bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521.

Chenzhuo Zhu, Song Han, Huizi Mao, and William J. Dally. 2017a. Trained ternary quantization. In *International Conference on Learning Representations (ICLR)*.

Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. 2017b. Trained ternary quantization. In *International Conference on Learning Representations (ICLR)*.

A Model Configuration

In Table 4, we list all configurations of CLIP models in our evaluation, including pre-trained CLIPs and knowledge distilled CLIPs.

B Training Hyperparameters

In Table 5, we exhibit hyperparameters used to train our TnCLIP model, while the values between square brackets are the to-be-selected hyperparameters and the values with the text bold font are the selected hyperparameters after tuning procedure.

		Model				
		CLIP LAION	**CLIP LAION**	**CLIP OpenAI**	**CLIP-KD**	**TnCLIP**
	Teacher Model	Pretrained	Pretrained	Pretrained	LAION(ViT-L/14)	LAION(ViT-L/14)
	Parameters	427.6M	149.6M	149.6M	149.6M	149.6M
	Embed_Dim	768	512	512	512	512
Vision	**Structure**	ViT-L/14	ViT-B/16	ViT-B/16	ViT-B/16	ViT-B/16
	Image Size	224	224	224	224	224
	Patch	14	16	16	16	16
	Layers	24	12	12	12	12
	Width	1024	768	768	768	768
Text	**Vocab Size**	49408	49408	49408	49408	49408
	Context	77	77	77	77	77
	Layers	12	12	12	12	12
	Width	768	512	512	512	512
	Heads	12	12	12	12	12

Table 4: Model configurations, including name, training strategy(whether to use distillation or not), number of parameters, embedding dimension, vision and text configurations.

model	Teacher: [**ViT-L/14**] Student: [**ViT-B/16**]
weight	[OpenAI_400M_ep32, **Laion400M_ep32**]
quantization	weight_quant: [**ternary**, int3, int4] activation_quant: [int8, **float16**]
Ternarization	β: [1, **2**, 3], ϵ: [**1e-6**]
precision	[**amp**, amp_bf16, bp16, fp32]
data load	num_workers: [4, 8, **16**, 32]
epoch	[32, **64**]
lr	[1e-4, 5e-4, **1e-3**]
warmup	[1000, 5000, **10000**, 100000]
batch size	[128 *8, 256 *8, **384** *8, 512 *8]
opt	adamw(**0.9**, [0.98, 0.998, **0.999**], ϵ: [**1e-6**])
wd	[0.01, 0.05, **0.1**, 0.2]
$\boldsymbol{\lambda}_{\text{crd}}, \boldsymbol{\tau_{crd}}$	[0.5, **1**, 2], [0.5, **1**, 2]
$\boldsymbol{\lambda}_{\text{icl}}$	[0.5, **1**, 2]
$\boldsymbol{\lambda_{fd}}$	[1000, **2000**, 4000]
augment cfg	[**None**, Scale, Scale+Color_Jitter+Gray_Scale]

Table 5: Training hyperparameters of TnCLIP: the value with text bold font is the selected hyperparameters after tuning. For example, $\lambda_{\text{crd}} = 1.0$ and temperature $\tau_{crd} = 1.0$.

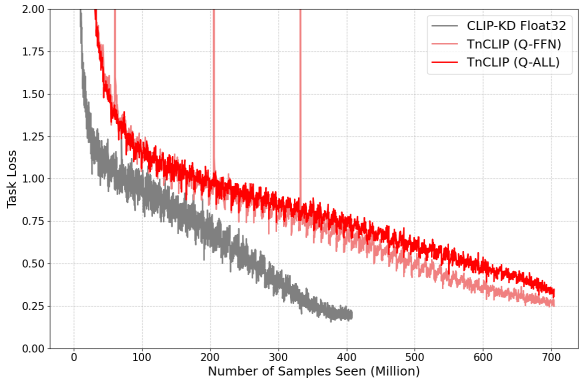

Figure 7: Task losses of full-precision CLIP-KD, ternary TnCLIP_Q-FFN and TnCLIP_Q-ALL.

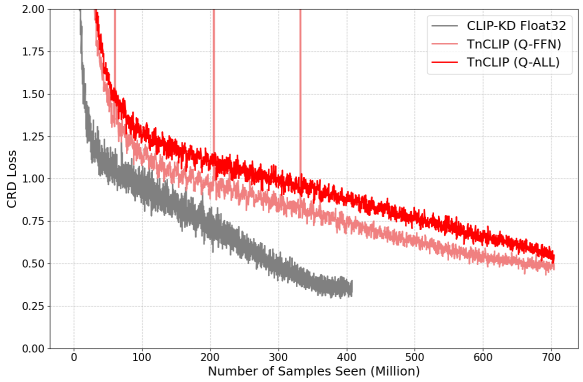

Figure 8: CRD losses of full-precision CLIP-KD, ternary TnCLIP_Q-FFN and TnCLIP_Q-ALL.

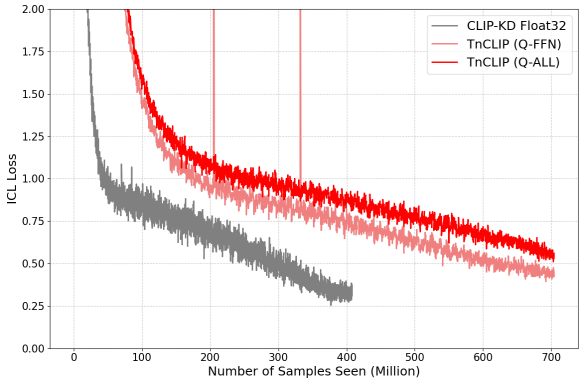

Figure 9: ICL losses of full-precision CLIP-KD, ternary TnCLIP_Q-FFN and TnCLIP_Q-ALL.

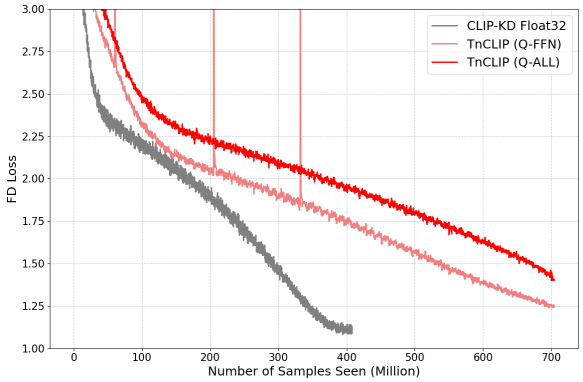

Figure 10: FD losses of full-precision CLIP-KD, ternary TnCLIP_Q-FFN and TnCLIP_Q-ALL.

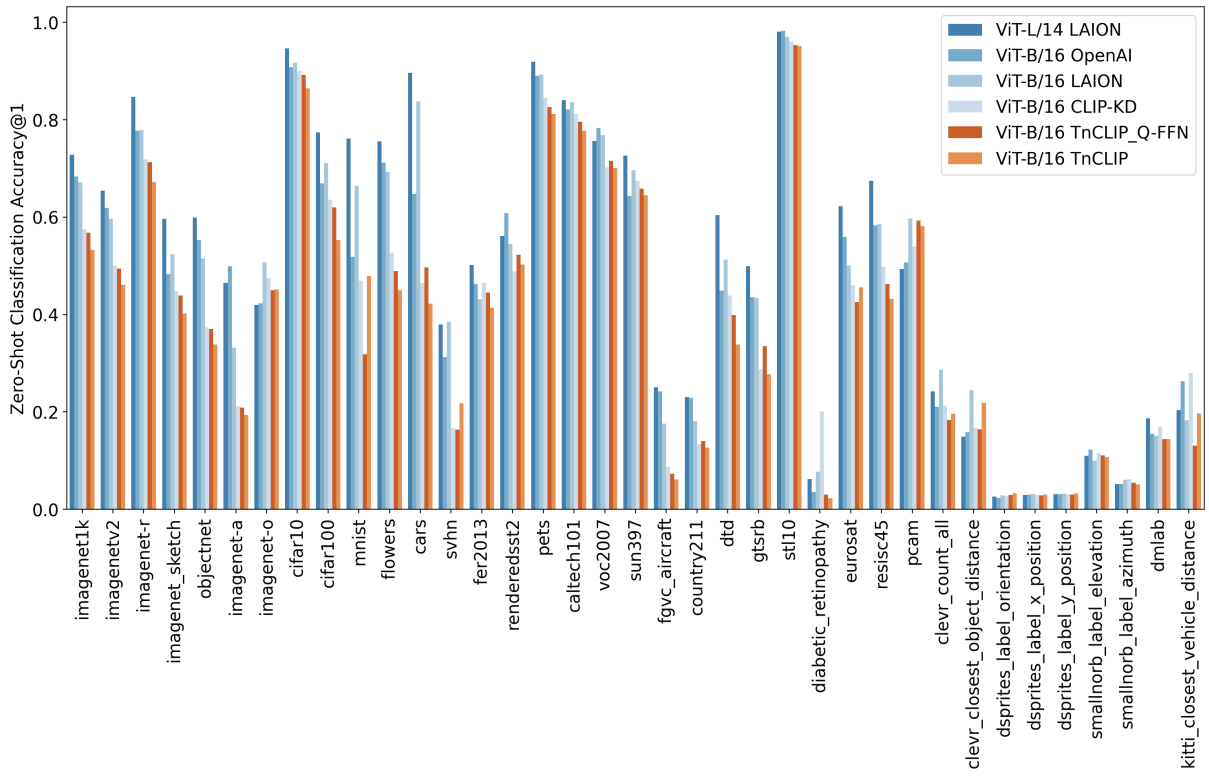

Figure 11: Zero-shot image classification performance: Accuracy@1 across 37 datasets.

C Details of Training TnCLIP

In Figure 7, 8, 9, and 10, we show the loss curves of three models on the number of samples seen. The first model is full-precision CLIP-KD, the second model is TnCLIP (Q-FFN) with FFN ternarized, and the third model is TnCLIP (Q-ALL) with MHA+FFN ternarized. The first loss curve is task loss. The second loss curve is contrastive relational distillation (CRD) loss. The third loss curve is interactive constrastive learning (ICL) loss. The last loss curve is feature distillation (FD) loss. The loss curves are plotted against the number of samples seen during training.

D Details of Zero-Shot Classification

In Figure 11 and Table 7, it is shown that TnCLIP with 99% ternary parameters obtains only 2.71% performance reduction compared with CLIP-KD of the same model architecture and size.

E Details of Inference Latency

In Table 6, we present a comprehensive analysis of CLIP model inference latency across various precision formats with the corresponding bit-per-weight (BPW) configurations, showing the decomposition of total latency into model loading, image loading, and model forwarding components, with all benchmarks conducted on Apple M4 Pro hardware over 1,000 test rounds.

F Ablation Study

In Figure 12 and Figure 13, we show the ablation study of data augmentation and int8 activation quantization. Based on the performance gap of loss curves, we determine training TnCLIP without any data augmentation and activation quantization.

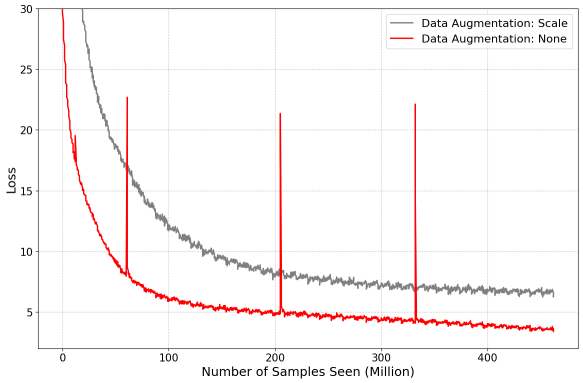

Figure 12: Ablation study of data augmentation on TnCLIP. Data augmentation makes loss convergence much more slower than the baseline.

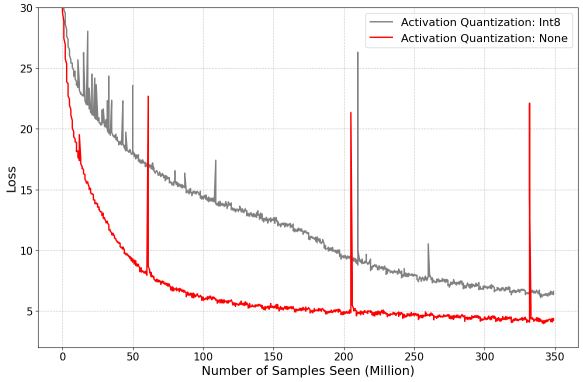

Figure 13: Ablation study of int8 or float16 activation quantization on TnCLIP. Int8 activation quantization makes loss convergence much more slower than the baseline.

Model	Precision	BPW ↓	Storage(MB) ↓	Model Load(ms) ↓	Image Load(ms) ↓	Model Forward(ms) ↓	Total Latency(ms) ↓
ViT-L/14	Float32	32	1630.9	382.22 ± 22.58	5.32 ± 1.65	498.60 ± 85.66	886.15 ± 90.65
	Float16	16	817.3	199.42 ± 9.50	5.35 ± 0.24	506.10 ± 75.54	710.88 ± 76.38
ViT-B/16	Float32	32	571.6	153.02 ± 7.41	5.35 ± 0.22	83.02 ± 55.47	241.40 ± 56.47
	Float16	16	287.7	90.15 ± 4.27	5.42 ± 0.63	79.04 ± 50.26	174.62 ± 50.52
TnCLIP_Q_FFN ViT-B/16	Float32	32	571.6	154.55 ± 21.00	5.79 ± 0.77	90.03 ± 59.39	250.38 ± 67.65
	Float16	16	287.7	90.52 ± 4.87	5.84 ± 0.72	86.62 ± 53.00	182.99 ± 53.57
	Q4_0	4	140.9	57.32 ± 4.35	5.83 ± 0.44	74.02 ± 49.97	137.18 ± 50.52
	Q4_1	4	147.3	58.53 ± 2.86	5.82 ± 0.38	70.40 ± 48.54	134.75 ± 48.68
	TQ1_0	1.6875	105	49.09 ± 3.47	5.81 ± 0.33	71.14 ± 50.89	126.05 ± 51.05
	TQ2_0	2.0625	109.8	50.15 ± 3.00	5.82 ± 0.38	76.26 ± 50.06	132.23 ± 49.99
TnCLIP ViT-B/16	Float32	32	571.6	149.82 ± 11.73	5.63 ± 0.45	87.83 ± 56.36	243.28 ± 57.14
	Float16	16	287.7	89.12 ± 3.77	5.77 ± 0.32	79.70 ± 50.35	174.60 ± 50.40
	Q4_0	4	84.9	44.16 ± 1.79	5.75 ± 0.24	71.30 ± 49.68	121.22 ± 49.79
	Q4_1	4	93.7	46.10 ± 1.87	5.75 ± 0.29	68.46 ± 46.07	120.32 ± 46.08
	TQ1_0	**1.6875**	**35.3**	**33.05 ± 1.70**	**5.74 ± 0.19**	**67.96 ± 45.58**	**106.75 ± 45.68**
	TQ2_0	2.0625	41.9	34.77 ± 2.75	5.78 ± 0.45	75.78 ± 48.16	116.33 ± 48.21

Table 6: CLIP model inference latency overview on different precisions and bpw (bits per weight). The CPU hardware is Apple M4 Pro. Total latency is combined with model loading, image loading and model forwarding. Latency = $A \pm B$: A represents the average value and B represents three standard deviations $3 * \sigma$ under 1,000 rounds of benchmarks.

Dataset	Type	ViT-L/14 LAION	ViT-B/16 OpenAI	ViT-B/16 LAION	ViT-B/16 CLIP-KD	ViT-B/16 TnCLIP_Q-FFN	ViT-B/16 TnCLIP
Natural Datasets							
caltech101	natural	83.99%	82.10%	83.63%	81.15%	79.61%	77.70%
cars	natural	89.64%	64.73%	83.77%	46.50%	49.67%	42.20%
cifar10	natural	94.63%	90.77%	91.73%	89.99%	89.24%	86.45%
cifar100	natural	77.39%	66.94%	71.15%	63.57%	61.96%	55.28%
country211	natural	23.04%	22.87%	18.12%	13.41%	14.04%	12.68%
dtd	natural	60.43%	44.95%	51.28%	43.88%	39.89%	33.83%
fer2013	natural	50.22%	46.22%	43.13%	46.53%	44.52%	41.40%
fgvc_aircraft	natural	25.02%	24.24%	17.64%	8.73%	7.29%	6.15%
flowers	natural	75.56%	71.18%	69.28%	52.66%	48.92%	44.97%
gtsrb	natural	49.92%	43.56%	43.42%	28.79%	33.49%	27.75%
imagenet-a	natural	46.49%	49.92%	33.19%	21.12%	20.91%	19.36%
imagenet-o	natural	41.95%	42.30%	50.65%	47.45%	45.00%	45.20%
imagenet-r	natural	84.68%	77.71%	77.93%	71.83%	71.28%	67.22%
imagenet1k	natural	72.77%	68.36%	67.07%	57.44%	56.76%	53.28%
imagenetv2	natural	65.41%	61.90%	59.65%	50.01%	49.43%	46.10%
objectnet	natural	59.86%	55.33%	51.49%	37.41%	37.02%	33.89%
pets	natural	91.91%	89.04%	89.26%	84.44%	82.56%	81.14%
stl10	natural	98.05%	98.25%	96.99%	96.05%	95.29%	95.16%
sun397	natural	72.59%	64.34%	69.61%	67.41%	65.84%	64.48%
svhn	natural	37.97%	31.27%	38.53%	16.72%	16.33%	21.79%
voc2007	natural	75.64%	78.32%	76.85%	70.23%	71.51%	70.10%
Specialized Datasets							
diabetic_retinopathy	specialized	6.19%	3.57%	7.75%	20.09%	3.03%	2.27%
eurosat	specialized	62.24%	56.00%	50.07%	45.96%	42.59%	45.59%
imagenet_sketch	specialized	59.65%	48.24%	52.37%	44.79%	43.91%	40.23%
mnist	specialized	76.10%	51.85%	66.39%	46.90%	31.86%	47.93%
pcam	specialized	49.38%	50.67%	59.73%	53.98%	59.30%	58.18%
renderedsst2	specialized	56.12%	60.79%	54.48%	48.82%	52.28%	50.30%
resisc45	specialized	67.43%	58.27%	58.54%	49.83%	46.21%	43.24%
Structured Datasets							
clevr_closest_object_distance	structured	14.91%	15.83%	24.51%	16.75%	16.40%	21.90%
clevr_count_all	structured	24.25%	21.03%	28.65%	21.27%	18.37%	19.65%
dmlab	structured	18.66%	15.50%	15.10%	16.92%	14.39%	14.40%
dsprites_label_orientation	structured	2.61%	2.34%	2.87%	2.71%	2.92%	3.28%
dsprites_label_x_position	structured	2.99%	3.00%	3.15%	2.93%	2.86%	3.04%
dsprites_label_y_position	structured	3.16%	3.11%	3.24%	3.16%	3.07%	3.28%
kitti_closest_vehicle_distance	structured	20.39%	26.30%	18.28%	27.99%	13.08%	19.69%
smallnorb_label_azimuth	structured	5.25%	5.18%	6.02%	6.20%	5.46%	5.16%
smallnorb_label_elevation	structured	11.00%	12.21%	9.96%	11.59%	11.11%	10.71%
Summary							
Average (natural)	natural	65.58%	60.68%	61.16%	52.16%	51.46%	48.86%
Average (specialized)	specialized	53.87%	47.06%	49.90%	44.34%	39.88%	41.11%
Average (structured)	structured	11.47%	11.61%	12.42%	12.17%	9.74%	11.24%
Average (All)	All Types	50.20%	46.17%	47.18%	40.95%	39.12%	38.24%
Performance vs. ViT-L/14 LAION	All Types	rel. 0%	-4.03%	-3.02%	-9.25%	-11.08%	-11.96%
Performance vs. ViT-B/16 OpenAI	All Types	+4.03%	rel. 0%	+1.01%	-5.22%	-7.05%	-7.93%
Performance vs. ViT-B/16 LAION	All Types	+3.02%	-1.01%	rel. 0%	-6.23%	-8.06%	-8.94%
Performance vs. ViT-B/16 CLIP-KD	All Types	+9.25%	+5.22%	+6.23%	rel. 0%	**-1.83%**	**-2.71%**

Table 7: Zero-shot classification performance: Accuracy@1 across 37 datasets.

G Ternary Weight Distribution

In Figure 14 15 16, we illustrate the distribution of ternary weight for TnCLIP Q-FFN and Q-ALL models. It is obvious that ternary weights maintain a good sparsity in addition to other benifits of quantization that we conlude during experiments.

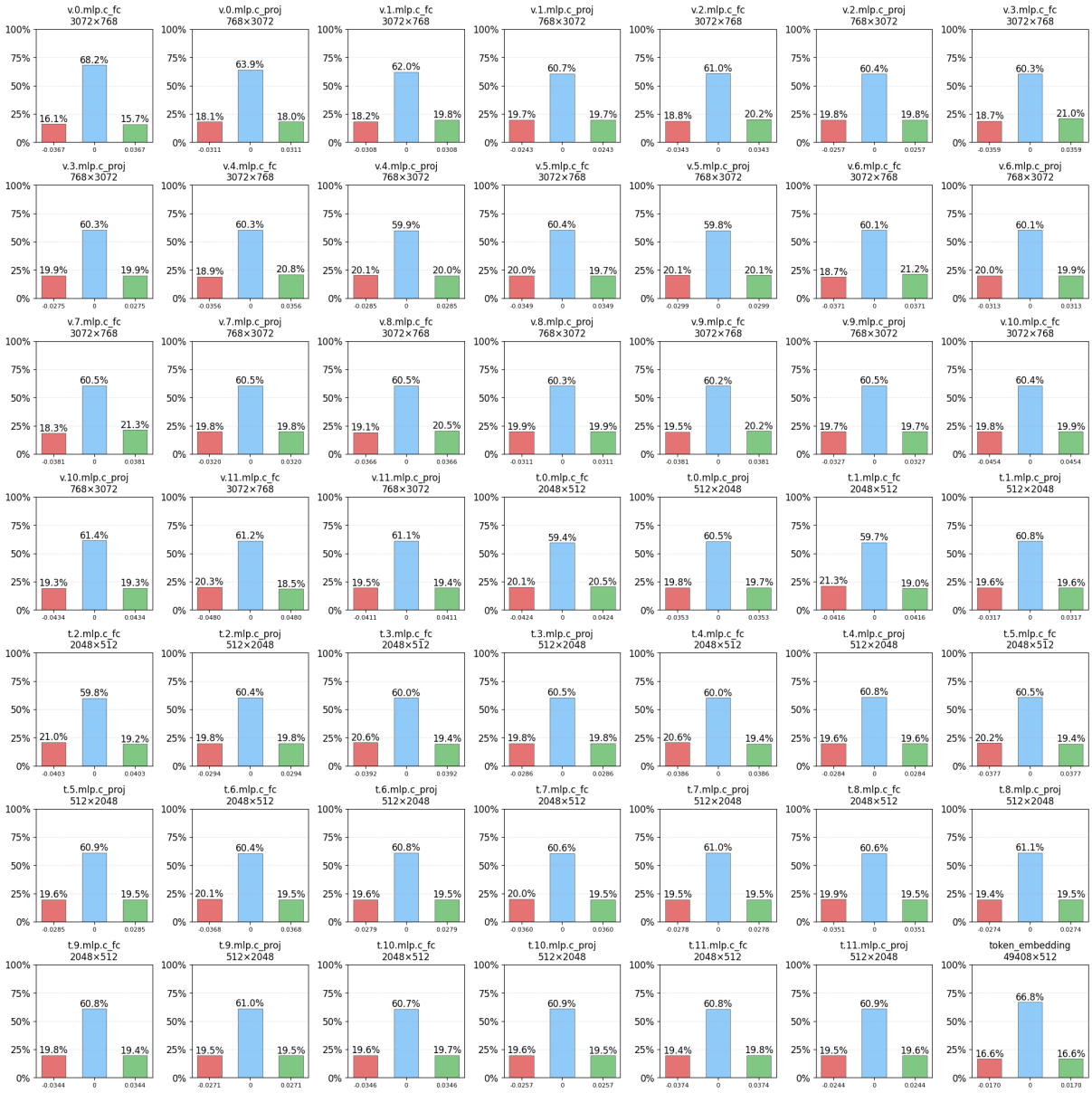

Figure 14: Ternary weight distribution of TnCLIP Q-FFN.

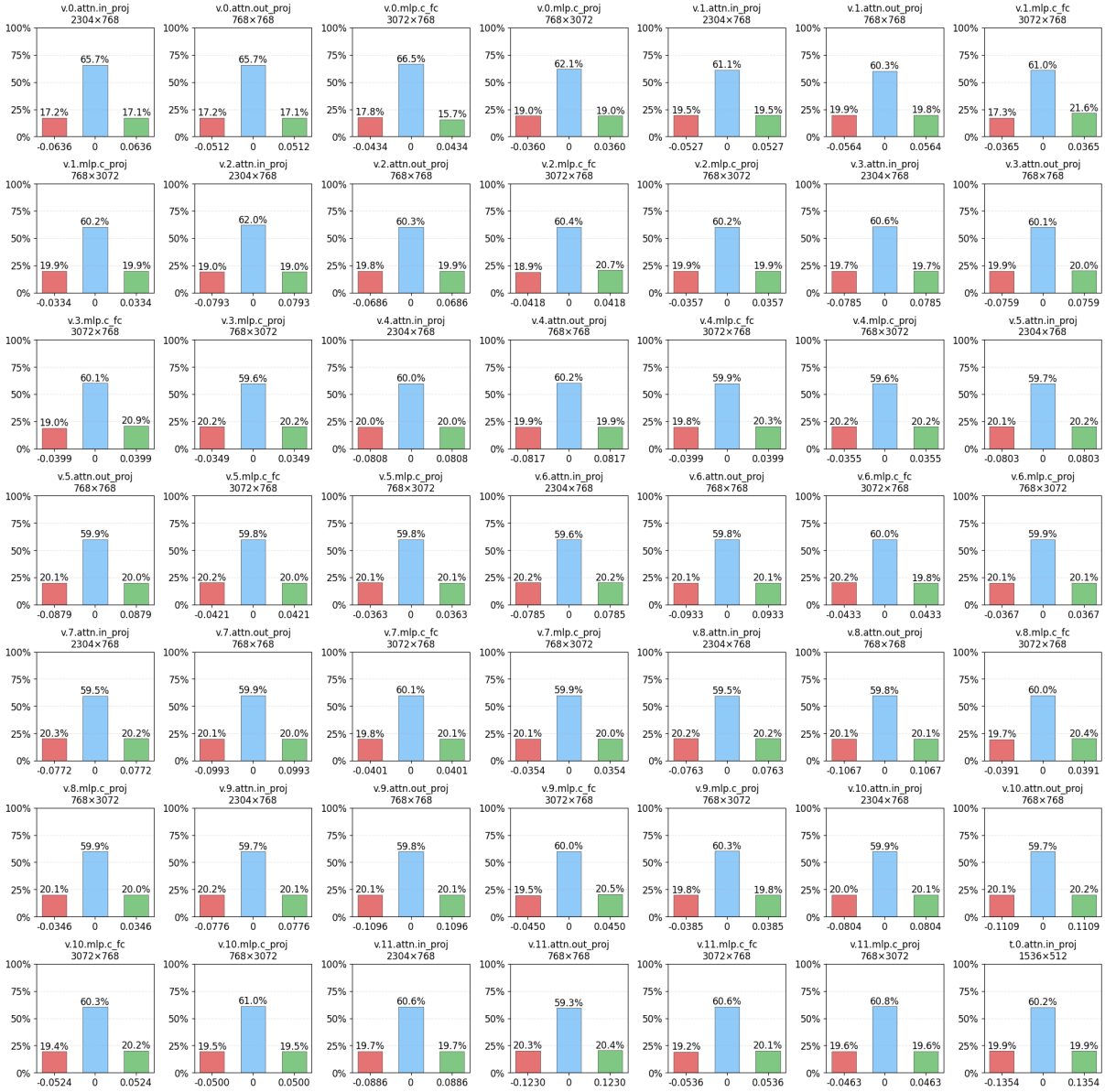

Figure 15: Ternary weight distribution of TnCLIP Q-ALL (Part1).

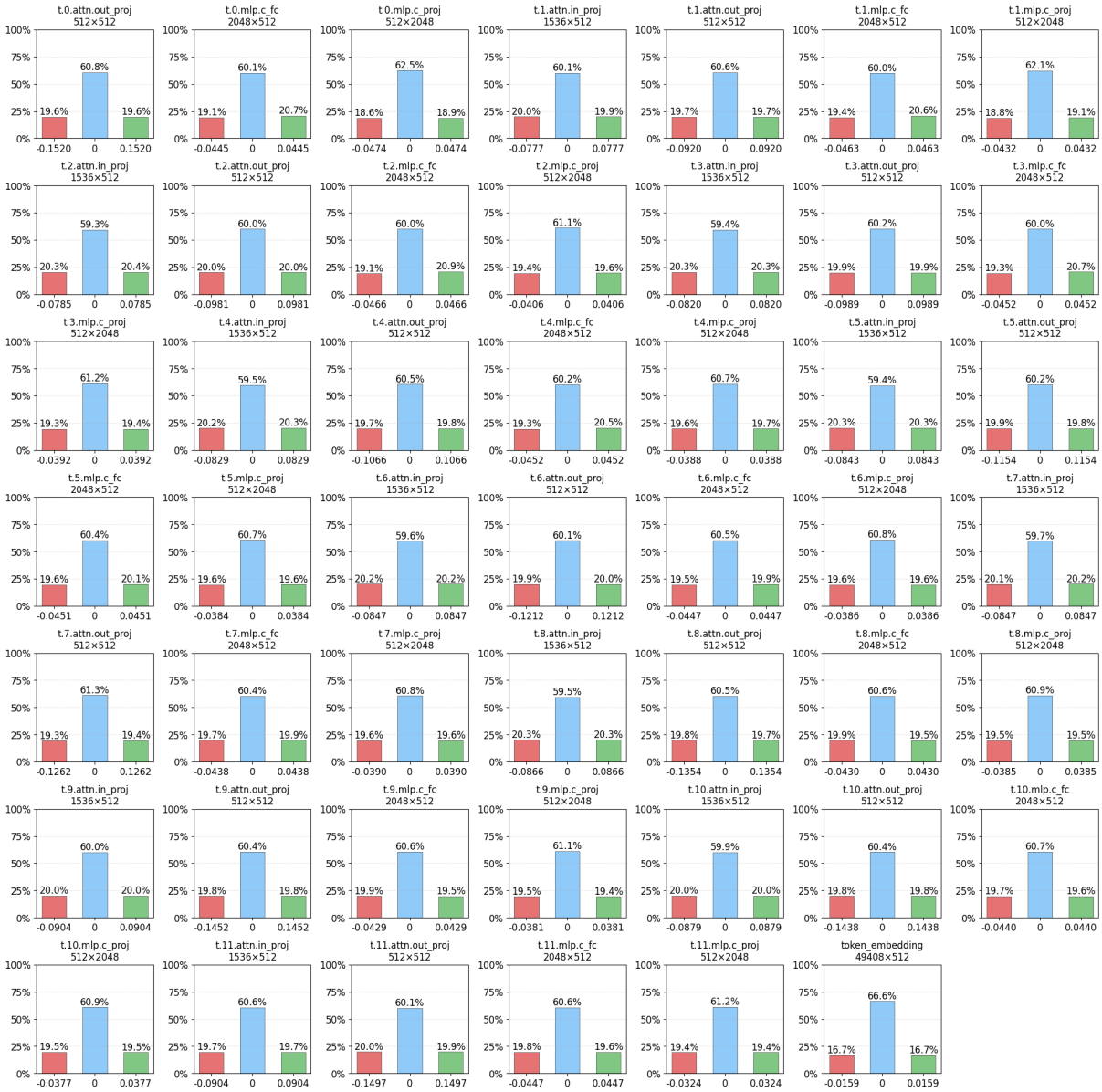

Figure 16: Ternary weight distribution of TnCLIP Q-ALL (Part2).