NEUCLIP: EFFICIENT LARGE-SCALE CLIP TRAINING WITH NEURAL NORMALIZER OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Accurately estimating the normalization term (also known as the partition function) in the contrastive loss is a central challenge for training Contrastive Language-Image Pre-training (CLIP) models. Conventional methods rely on large batches for approximation, demanding substantial computational resources. To mitigate this issue, prior works introduced per-sample normalizer estimators, which are updated at each epoch in a blockwise coordinate manner to keep track of updated encoders. However, this scheme incurs optimization error that scales with the ratio of dataset size to batch size, limiting effectiveness for large datasets or small batches. To overcome this limitation, we propose NeuCLIP, a novel and elegant optimization framework based on two key ideas: (i) **reformulating** the contrastive loss for each sample **via convex analysis** into a minimization problem with an auxiliary variable representing its log-normalizer; and (ii) **transforming** the resulting minimization over n auxiliary variables (where n is the dataset size) via **variational analysis** into the minimization over a compact neural network that predicts the log-normalizers. We design an alternating optimization algorithm that jointly trains the CLIP model and the auxiliary network. By employing a tailored architecture and acceleration techniques for the auxiliary network, NeuCLIP achieves more accurate normalizer estimation, leading to improved performance compared with previous methods. Extensive experiments on large-scale CLIP training, spanning datasets from millions to billions of samples, demonstrate that NeuCLIP outperforms previous methods.

1 INTRODUCTION

Since its introduction, Contrastive Language-Image Pretraining (CLIP) (Radford et al., 2021) has emerged as the de facto standard for vision-language representation learning. The strong capability to align images with corresponding texts has made CLIP valuable for a wide range of real-world applications, including zero-shot classification (Qian & Hu, 2024), cross-modal retrieval (Zeng & Mao, 2022), text-to-image generation (Ramesh et al., 2022), and high-quality dataset selection (Fang et al., 2023a). With the rise of large language models (LLMs), CLIP has also been widely adopted to equip LLMs with the ability to interpret visual inputs (Bai et al., 2025).

A fundamental impediment to training CLIP models is their extensive dependence on huge datasets: attaining competitive performance typically mandates access to millions or even billions of image-text pairs (Fang et al., 2023a; Wang et al., 2025b). A mainstream approach for training CLIP models is to optimize a bimodal contrastive loss, which contrasts each positive image-text pair against numerous negative pairs. To enable training on billions of samples, two primary strategies have emerged to approximate the prohibitive normalization term required for contrastive loss gradient calculation. **The first strategy**, which relies on massive GPU resources, uses an extremely large batch size to construct a contrastive loss within each batch for backpropagation. This strategy is used by many works, including OpenAI CLIP (Radford et al., 2021) and OpenCLIP (Cherti et al., 2023). **The second strategy** addresses the high resource demand of the first by directly optimizing a global contrastive loss, which contrasts each positive pair with all negative pairs. To tackle the computational challenge, an estimator of the normalizer for each sample's contrastive loss is maintained and updated using a moving average formula following a rigorous framework of finite-sum coupled compositional optimization. It was first proposed by Yuan et al. (2022) for unimodal contrastive self-supervised learning, and later adopted by Wei et al. (2024) with significant improvements for

CLIP training, yielding the FastCLIP method. While being less resource demanding, this strategy suffers an inherent limitation that the optimization error scales with the ratio of dataset size to batch size, constraining its effectiveness for large datasets or small batch sizes.

Recently, there emerge some new ideas for CLIP training. For example, Zhai et al. (2023) proposed SigCLIP with a sigmoid-based contrastive loss by formulating the problem as a binary classification problem, which avoids the computation of the normalization term involving numerous negative pairs. However, SigCLIP still requires a large batch size to achieve competitive performance. Sun et al. (2025) proposed AmorLIP that leverages a lightweight network to predict the normalizer of each contrastive loss. While conceptually appealing, AmorLIP's objective for training the lightweight network still faces the challenge of estimating the log-partition function for the gradient calculation of the lightweight network, leading to a chicken-and-egg problem.

This paper aims to address the limitations of FastCLIP and AmorLIP for CLIP training through a principled approach towards optimizing the global contrastive loss with a neural normalizer. Our method is built on two key ideas: (1) using convex analysis, we reformulate the contrastive loss of each anchor data to a minimization problem with an auxiliary variable, whose optimal solution corresponds to the log-normalizer; and (2) using variational analysis, we transform the minimization over n auxiliary variables (where n is dataset size) into minimization over a compact network that directly predicts the log-normalizers, referred to as the **normalizer-prediction network (NPN)**.

Compared with FastCLIP and AmorLIP, our method offers several notable advantages. First, the objective for learning the encoders and the NPN is unified, and its gradient avoids any nonlinear dependence on the partition function. This allows traditional stochastic gradient methods to be employed for updating both the encoders and the NPN without incurring gradient estimation bias. Second, the unrestricted optimal solution of the auxiliary variable motivates us to inject inductive bias into the design of the NPN, resulting in a simple yet effective architecture: a feedforward layer on top of the encoders followed by a log-sum-exponential pooling layer. Moreover, this seamless optimization framework enables key acceleration techniques, including alternating optimization with multiple NPN updates and periodic re-initialization of the NPN's parameters using sampled updated embeddings, which yields significantly better normalizer approximation. The **contributions** of this paper are threefold:

- We reformulate the contrastive loss into an equivalent form in which the normalization terms are explicitly exposed as optimization variables. This reformulation provides a principled foundation for efficient neural normalizer approximation.
- We introduce a joint optimization problem that learns the encoders and a compact normalizer-prediction network (NPN) with a unified objective, which is derived from variational analysis. We also develop an efficient algorithm for alternatively optimizing the NPN and the CLIP encoders.
- We validate the effectiveness of our approach through extensive experiments on large-scale datasets, showing consistent improvement over existing methods. Comprehensive ablation studies are also conducted to highlight the contribution of different components in our framework.

2 Related Works

Efficient Training of CLIP Models. Numerous approaches have been proposed to enhance the efficiency of CLIP model training. Prior works include curating high-quality datasets (Schuhmann et al., 2022; Fang et al., 2023a; Xu et al., 2024; Wang et al., 2024), designing more efficient vision encoder architectures (Fang et al., 2023b; Alabdulmohsin et al., 2023; Chen et al., 2024), and applying image-token masking to reduce computational cost (Li et al., 2023b;a). Additional strategies leverage knowledge distillation to train compact student models (Vasu et al., 2024) or employ a pretrained reference model to steer and accelerate the training of a target model, thereby improving scaling laws (Wei et al., 2025). In contrast, our work is orthogonal to these directions: we focus on improving the optimization process itself by providing a more efficient and stable method for minimizing the contrastive loss.

Optimizing the Global Contrastive Loss. The global contrastive loss was first introduced by Yuan et al. (2022) to address the large batch-size requirement of SimCLR (Chen et al., 2020). They proposed an efficient optimization algorithm, SogCLR, with provable convergence guarantees for unimodal self-supervised contrastive learning. Notably, SogCLR with a batch size of 256 matches

the performance of SimCLR with a batch size of 8192 on ImageNet. Subsequent work by Qiu et al. (2023) provided a distributionally robust optimization (DRO) interpretation of the global contrastive loss, leading to a constrained DRO formulation with individualized temperature optimization. Building on this perspective, Wei et al. (2024) proposed a simplified variant that unifies the temperature parameters into a single scalar. Another line of work explains the global contrastive loss from a probabilistic perspective, showing it as the maximum likelihood estimation of a discriminative model (Wang et al., 2025a). Despite these different viewpoints, all methods rely on optimization techniques similar to SogCLR, which maintain and update per-sample estimators for the normalization term of the contrastive loss. Consequently, they share a key limitation: the optimization error scales with the ratio between the dataset size and the batch size.

Learning with Auxiliary Networks. Leveraging auxiliary networks to facilitate training has been widely explored (Shen et al., 2024; He et al., 2020; Su et al., 2025; Kim et al., 2024; Evans et al., 2025; Qiu et al., 2024; Sun et al., 2025). We highlight two closely related works (Qiu et al., 2024; Sun et al., 2025). Qiu et al. (2024) introduced *TempNet*, a network designed to predict personalized temperatures for each sample when training CLIP models with a robust global contrastive loss. Their approach was also motivated by variational analysis that led to the joint optimization of encoders and TempNet. Nevertheless, their optimization algorithm, built on SogCLR, still requires maintaining and updating per-sample estimators of the contrastive loss normalization term, and thus inherits the same limitations of SogCLR. Sun et al. (2025) proposed *AmorLIP*, which optimizes a robust global contrastive loss similar to that of Wei et al. (2024) by jointly learning a lightweight network to approximate the partition function. However, AmorLIP differs from our method in several key aspects: (i) the objective for training the lightweight network is heuristically defined as divergence minimization between its predictions and the true partition function values, which still involves a non-linear function of the normalizer, leading to the chick-and-egg problem; (ii) AmorLIP simply employs a Multi-Layer Perceptron (MLP) with few layers for the NPN, while our design leverages inductive bias to improve performance, as validated in ablation studies. Furthermore, AmorLIP requires maintaining an exponential moving average (EMA) network of NPN to mitigate the chicken-and-egg issue, since its auxiliary objective still involves estimating a nonlinear function of the normalizer.

3 PRELIMINARY

Notations. We denote by $\boldsymbol{w} \in \mathbb{R}^d$ the parameters of the CLIP model. Let $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{z}_i)\}_{i=1}^n$ be a training dataset of n samples, where $\boldsymbol{x}_i$ is an image and $\boldsymbol{z}_i$ is its corresponding text description. The features of image $\boldsymbol{x}_i$ and text $\boldsymbol{z}_j$ output by the CLIP model are denoted by $\boldsymbol{e}_{1,i} = e_1(\boldsymbol{w}; \boldsymbol{x}_i) \in \mathbb{R}^d$ and $\boldsymbol{e}_{2,j} = e_2(\boldsymbol{w}; \boldsymbol{z}_j) \in \mathbb{R}^d$ respectively, where $e_1(\boldsymbol{w}; \cdot)$ and $e_2(\boldsymbol{w}; \cdot)$ denote the image encoder and the text encoder, respectively. The cosine similarity between features $\boldsymbol{e}_{1,i}$ and $\boldsymbol{e}_{2,j}$ is denoted as $s_{i,j} := \cos(\boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,j})$.

The convex conjugate of a function $f : \mathbb{R} \mapsto \mathbb{R}$ is given by $f^*(y) := \max_x y \cdot x - f(x)$. From the Fenchel–Moreau theorem (Rockafellar, 1997, Theorem 12.2) we know that if f is a proper, lower semi-continuous and convex function, the convex conjugate of f^* is equivalent to f, i.e., $f(x) = f^{**}(x) := \max_y x \cdot y - f^*(y)$.

Global Contrastive Loss. Following (Wei et al., 2024), we consider optimizing a robust global contrastive loss defined below:

$$\min_{\boldsymbol{w}, \tau \geq \tau_0} \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \log\Bigg(\varepsilon + \underbrace{\frac{1}{|\mathcal{S}|-1} \sum_{\boldsymbol{z}_j \in \mathcal{S}, j \neq i} \exp\left(\frac{s_{i,j} - s_{i,i}}{\tau}\right)}_{g_1(\boldsymbol{w}, \tau; i, \mathcal{S})}\Bigg) + \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{z}_i \in \mathcal{S}} \log\Bigg(\varepsilon + \underbrace{\frac{1}{|\mathcal{S}|-1} \sum_{\boldsymbol{x}_j \in \mathcal{S}, j \neq i} \exp\left(\frac{s_{j,i} - s_{i,i}}{\tau}\right)}_{g_2(\boldsymbol{w}, \tau; i, \mathcal{S})}\Bigg) + 2\tau\rho, \tag{1}$$

where τ is the temperature parameter, τ_0, ε are small constants, and $\rho > 0$ is a hyperparameter. In order to solve this problem, we need to compute an estimator of the gradient. In particular, the

gradient w.r.t. $\boldsymbol{w}$ is given by

$$\tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \frac{1}{\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})} \cdot \nabla g_1(\boldsymbol{w}, \tau; i, \mathcal{S}) + \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{z}_i \in \mathcal{S}} \frac{1}{\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{S})} \cdot \nabla g_2(\boldsymbol{w}, \tau; i, \mathcal{S}). \tag{2}$$

We can see that the terms $\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})$ and $\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{S})$ serve as **normalizers** of the gradient calculation for image $\boldsymbol{x}_i$ and text $\boldsymbol{z}_i$. A key challenge is that $g_1(\boldsymbol{w}, \tau; i, \mathcal{S})$ and $g_2(\boldsymbol{w}, \tau; i, \mathcal{S})$ cannot be computed exactly, as they depend on all other data samples. This necessitates approximating these normalizers using only a batch of samples.

Mini-batch Approximation. A naive approach is to simply use a mini-batch approximation (Radford et al., 2021; Cherti et al., 2023), i.e., sampling a subset $\mathcal{B} \subset \mathcal{S}$ and computing the following gradient estimator

$$\tau \cdot \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{x}_i \in \mathcal{B}} \frac{1}{\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{B})} \cdot \nabla g_1(\boldsymbol{w}, \tau; i, \mathcal{B}) + \tau \cdot \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{z}_i \in \mathcal{B}} \frac{1}{\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{B})} \cdot \nabla g_2(\boldsymbol{w}, \tau; i, \mathcal{B}).$$

This is equivalent to performing backpropagation on a mini-batch contrastive loss $\tau \cdot \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{B})) + \tau \cdot \log(\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{B}))$. However, this gradient estimator is biased as its expectation does not give the true gradient in Equation (2) due to the non-linearity of the reciprocal function. As a consequence, it requires a large batch size (Yuan et al., 2022).

Moving-average Approximation. To address the large batch issue, Yuan et al. (2022) proposed the SogCLR algorithm, which maintains two sequences of estimators $\{u_{1,i}^{(t)}, u_{2,i}^{(t)}\}$ for each $\boldsymbol{x}_i, \boldsymbol{z}_i$ to approximate $\varepsilon + g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{S})$ and $\varepsilon + g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{S})$ at the t-th iteration, respectively. At t-th iteration with solutions $(\boldsymbol{w}^{(t)}, \tau^{(t)})$, for $(\boldsymbol{x}_i, \boldsymbol{z}_i)$ in the sampled batch $\mathcal{B}^{(t)}$, their normalizer estimators are updated as follows

$$\begin{aligned} u_{1,i}^{(t+1)} &= (1-\gamma) u_{1,i}^{(t)} + \gamma(\varepsilon + g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)})), \\ u_{2,i}^{(t+1)} &= (1-\gamma) u_{2,i}^{(t)} + \gamma(\varepsilon + g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)})), \end{aligned} \tag{3}$$

where $\gamma \in [0, 1]$ is a hyperparameter. Then, the gradient estimator for $\boldsymbol{w}^{(t)}$ is computed by

$$\frac{\tau^{(t)}}{|\mathcal{B}^{(t)}|} \sum_{\boldsymbol{x}_i \in \mathcal{B}^{(t)}} \frac{1}{u_{1,i}^{(t+1)}} \cdot \nabla_{\boldsymbol{w}} g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}) + \frac{\tau^{(t)}}{|\mathcal{B}^{(t)}|} \sum_{\boldsymbol{z}_i \in \mathcal{B}^{(t)}} \frac{1}{u_{2,i}^{(t+1)}} \cdot \nabla_{\boldsymbol{w}} g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}). \tag{4}$$

It has been shown that the optimization error of SogCLR converges to zero (Yuan et al., 2022). Built on this idea, Wei et al. (2024) proposed FastCLIP, an efficient distributed CLIP training framework with several improvements including the temperature optimization and the learning rate schedule for γ. However, the convergence error of FastCLIP suffers from a scaling factor of $O(n/B)$ on the standard rate (Yuan et al., 2022), where $B = |\mathcal{B}^{(t)}|$ is the mini-batch size per-iteration. This property is not desirable since the error will increase when n increases and B decreases.

4 NeuCLIP: CLIP training with Neural Normalizer Optimization

In this section, we first present a reformulation of the contractive loss as a minimization problem. Then we derive a joint optimization problem from variational analysis to learn the encoders and the normalizer-prediction network (NPN). Finally, we present an optimization algorithm.

4.1 Reformulating the contrastive loss

Without lose of generality, let us consider the individual contrastive loss for an image anchor data $\boldsymbol{x}_i$, as given by $F(\boldsymbol{w}, \tau; \boldsymbol{x}_i) = \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S}))$. Since $f(\cdot) = -\log(\cdot)$ is a convex function, we can leverage the conjugate transformation $f(x) = \max_y y \cdot x - f^*(y)$ with $f^*(y) = -\log(-y) - 1$ to reformulate the above individual contrastive loss as follows (by setting $x = \varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})$):

$$\begin{aligned} &F(\boldsymbol{w}, \tau; \boldsymbol{x_i}) = \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) = -\max_y \{y \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) - f^*(y)\} \\ &= -\max_y \{y \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \log(-y) + 1\} = \min_y \{-y \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) - \log(-y) - 1\} \\ &= \min_\alpha \{\exp(-\alpha) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha - 1\}, \end{aligned} \tag{5}$$

where the last equality uses a change of variable $\alpha = -\log(-y)$. It is not difficult to derive that the optimal solution α^* to the last optimization problem is given by $\alpha^* = \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S}))$ (c.f. Appendix A.1), which is exactly the log-normalizer.

Substituting the above reformulation of each contrastive loss in Equation (1), we get the following equivalent form of the global contrastive loss:

$$\begin{aligned}\min_{\boldsymbol{w},\tau} \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \left\{ \min_{\alpha_{1,i}} \exp(-\alpha_{1,i}) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_{1,i} - 1 \right\} \\ + \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{z}_i \in \mathcal{S}} \left\{ \min_{\alpha_{2,i}} \exp(-\alpha_{2,i}) \cdot (\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_{2,i} - 1 \right\} + 2\tau\rho.\end{aligned} \tag{6}$$

Indeed, the update of u_1, u_2 sequences of SogCLR in Equation (3) can be recovered by solving the above problem using stochastic block mirror descent method (Lan, 2020, Section 4.6.2). We provide a detailed derivation in Appendix A.2.

4.2 Neural Normalizer Optimization

Maintaining and updating $\{\alpha_{1,i}, \alpha_{2,i}\}$ in a coordinate-wise manner is the root that leads to a scaling factor of $O(n/B)$ in the convergence error of FastCLIP. To mitigate this issue, our idea is grounded in the following theorem from variational analysis.

Theorem 1 (Rockafellar & Wets, 2009, Theorem 14.60)**.** *Let $\mathcal{F}$ be a space of measurable functions from Ω to $\mathbb{R}$ that is decomposable relative to a finite measure μ. Let $f : \Omega \times \mathbb{R} \to \mathbb{R}$ be a normal integrand. Then, as long as $\int_{x\in\Omega} f(x, \alpha(x))\mu(dx) \neq \infty$ for all $\alpha(\cdot) \in \mathcal{F}$, we have*

$$\inf_{\alpha(\cdot)\in\mathcal{F}} \int_{x\in\Omega} f(x, \alpha(x))\mu(dx) = \int_{x\in\Omega} \left(\inf_{\alpha\in\mathbb{R}} f(x, \alpha) \right) \mu(dx). \tag{7}$$

Moreover, if the above infimum is not $-\infty$, then $\alpha^*(\cdot) \in \arg\min_{\alpha(\cdot)\in\mathcal{F}} \int_{x\in\Omega} f(x, \alpha(x))\mu(dx)$ if and only if $\alpha^*(x) \in \arg\min_{\alpha\in\mathbb{R}} f(x, \alpha)$ for μ-almost every $x \in \Omega$.

The above equality indicates that the minimization over individual variables α for each x on the right hand side within an integral of x can be translated into searching for a function $\alpha(\cdot) \in \mathcal{F}$ that minimizes the whole integral over all $x \in \Omega$.

Our reformulated contrastive loss (Equation 6) shares a similar structure to the right hand side of Equation (7) when the measure μ is a probability measure, with minimization over $\alpha_{1,i}, \alpha_{2,i}$, and then the average over all samples. Hence, Theorem 1 implies that

$$\begin{aligned}&\tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \left\{ \min_{\alpha_{1,i}} \exp(-\alpha_{1,i}) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_{1,i} - 1 \right\} \\ &= \min_{\alpha_1(\cdot)\in\mathcal{F}} \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \left\{ \exp(-\alpha_1(\boldsymbol{x}_i)) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_1(\boldsymbol{x}_i) - 1 \right\}.\end{aligned}$$

As a result, Equation (6) can be transformed into:

$$\begin{aligned}\min_{\boldsymbol{w},\tau} \min_{\alpha_1(\cdot),\alpha_2(\cdot)\in\mathcal{F}} \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} \left\{ \exp(-\alpha_1(\boldsymbol{x}_i)) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_1(\boldsymbol{x}_i) - 1 \right\} \\ + \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{z}_i \in \mathcal{S}} \left\{ \exp(-\alpha_2(\boldsymbol{z}_i)) \cdot (\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_2(\boldsymbol{z}_i) - 1 \right\} + 2\rho\tau.\end{aligned} \tag{8}$$

Neural Normalizer Optimization. Directly solving (8) is not easier than solving (6) due to the constraint $\alpha_1(\cdot), \alpha_2(\cdot) \in \mathcal{F}$. Our strategy is to approximate these functions using parameterized neural networks. Specifically, we solve the problem by restricting $\alpha_1(\cdot) \in \mathcal{F}_{\boldsymbol{W}_1}$ and $\alpha_2(\cdot) \in \mathcal{F}_{\boldsymbol{W}_2}$, where $\mathcal{F}_{\boldsymbol{W}_1}$ and $\mathcal{F}_{\boldsymbol{W}_2}$ denote the function classes represented by neural networks parameterized by $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$, respectively. This raises the question of how to design the architecture of the neural network. A naive idea is to use simple feedforward neural networks. It is guaranteed by universal

approximation theory that a neural network can approximate any continuous function arbitrarily well as long as the network is wide enough. However, this could increase the burden of training. Instead, we draw insights from Theorem 1 to design a network with inductive bias, which implies

$$\alpha_1(\boldsymbol{x}_i) \in \arg\min_{\alpha_{1,i}} \exp(-\alpha_{1,i}) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_{1,i} - 1$$
$$= \log\left(\varepsilon + \frac{1}{|\mathcal{S}| - 1} \sum_{\boldsymbol{z}_j \in \mathcal{S}, j \neq i} \exp\left(\frac{\boldsymbol{e}_{1,i}^T \boldsymbol{e}_{2,j} - \boldsymbol{e}_{1,i}^T \boldsymbol{e}_{2,i}}{\tau}\right)\right), \tag{9}$$

where the last equality is from Equation (5) and the definition of g. Since $\boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i}$ are readily available from the CLIP encoders, we only need a model that compresses the information of all $\boldsymbol{e}_{2,j}$. Inspired by this, we define the following network architecture with parameter $\boldsymbol{W}_1 \in \mathbb{R}^{d \times m}$, where m is the number of neurons of the hidden layer:

$$\alpha_1(\boldsymbol{x}_i) := \alpha_1(\boldsymbol{W}_1; \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i}) = \log\left(\varepsilon + \frac{1}{m} \sum_{j'=1}^{m} \exp\left(\frac{\cos(\boldsymbol{e}_{1,i}, \boldsymbol{W}_{1,j'}) - \boldsymbol{e}_{1,i}^T \boldsymbol{e}_{2,i}}{\tau}\right)\right), \tag{10}$$

where $\boldsymbol{W}_{1,j'}$ denotes the j'-th column of $\boldsymbol{W}_1$. This can be seen as a compact network built on top of the encoders, processing their output embeddings $\{\boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i}\}$ with a feedforward layer parameterized by $\boldsymbol{W}_1$ and followed by a log-sum-exponential pooling layer. Compared with the unrestricted optimal solution of $\alpha(\boldsymbol{x}_i)$ in Equation (9), we can view $\boldsymbol{W}_{1,1}, \ldots, \boldsymbol{W}_{1,m}$ as prototypical embeddings that summarize $\{\boldsymbol{z}_j\}$. This is supported by existing studies of self-supervised representation learning, which show that the learned embeddings of training samples from the same class tend to concentrate around their class means (Ben-Shaul et al., 2023). Similarly, we use the following network with an additional parameter $\boldsymbol{W}_2 \in \mathbb{R}^{d \times m}$ to approximate $\alpha_2(\boldsymbol{z}_i)$ by

$$\alpha_2(\boldsymbol{z}_i) := \alpha_2(\boldsymbol{W}_2; \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i}) = \log\left(\varepsilon + \frac{1}{m} \sum_{j'=1}^{m} \exp\left(\frac{\cos(\boldsymbol{e}_{2,i}, \boldsymbol{W}_{2,j'}) - \boldsymbol{e}_{1,i}^T \boldsymbol{e}_{2,i}}{\tau}\right)\right). \tag{11}$$

Finally, our unified objective for learning the encoders and the NPN becomes

$$\min_{\boldsymbol{w}, \tau, \boldsymbol{W}_1, \boldsymbol{W}_2} \tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{x}_i \in \mathcal{S}} (\exp(-\alpha_1(\boldsymbol{W}_1, \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i})) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_1(\boldsymbol{W}_1, \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i})) +$$
$$\tau \cdot \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{z}_i \in \mathcal{S}} (\exp(-\alpha_2(\boldsymbol{W}_2, \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i})) \cdot (\varepsilon + g_2(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha_2(\boldsymbol{W}_2, \boldsymbol{e}_{1,i}, \boldsymbol{e}_{2,i})) + 2\tau(\rho - 1). \tag{12}$$

4.3 Alternating optimization and acceleration

To solve Equation (12), a straightforward approach is to update $\boldsymbol{w}, \tau, \boldsymbol{W}_1, \boldsymbol{W}_2$ simultaneously by using stochastic gradient based methods. However, we find that this approach does not work well in practice (see Appendix B.3 for empirical results). The reasons are multi-fold: (i) the overall objective landscape of $\boldsymbol{w}, \tau, \boldsymbol{W}_1, \boldsymbol{W}_2$ is much more complicated than the original objective in terms of $\boldsymbol{w}, \tau$; (ii) the NPNs' predictions also rely on the output embeddings of the encoders, which makes the predicted normalizers from one step update of $\boldsymbol{W}_1, \boldsymbol{W}_2$ not good enough for updating the parameters $\boldsymbol{w}, \tau$. A natural idea to address this issue is to split the parameters $\boldsymbol{w}, \tau, \boldsymbol{W}_1, \boldsymbol{W}_2$ into two blocks $(\boldsymbol{w}, \tau)$ and $(\boldsymbol{W}_1, \boldsymbol{W}_2)$, and use an alternating optimization scheme to update two blocks one by one. A similar strategy has been used for other problems that exhibit two natural blocks, e.g., non-negative matrix factorization (Lin, 2007).

A straightforward optimization scheme is to alternate the optimization over $\boldsymbol{W}_1, \boldsymbol{W}_2$ given $\boldsymbol{w}, \tau$ and then the optimization over $\boldsymbol{w}, \tau$ given $\boldsymbol{W}_1, \boldsymbol{W}_2$, which is proved to enjoy a convergence guarantee (Grippof & Sciandrone, 1999, Theorem 6.3). However, it is not implementable as exactly solving the optimization problem over one block given another block is unrealistic. To address this, we present a practical method in Algorithm 1, which is referred to as NeuCLIP. For comparison, we present FastCLIP in Algorithm 2.

Algorithm 1: The NeuCLIP Algorithm

Input: CLIP model $\boldsymbol{w}^{(0)}$, temperature $\tau^{(0)}$, NPNs $\boldsymbol{W}_1^{(0)}, \boldsymbol{W}_2^{(0)}$, dataset $\mathcal{S}$, number of iterations T, restart frequency T_r and number of updates T_u for NPNs

for $t = 0, \ldots, T-1$ **do**
- Randomly sample a mini-batch $\mathcal{B}^{(t)} \subset \mathcal{S}$;
- **if** $t \mod T_r = 0$ **then** `// Restart`
  - Reset $\boldsymbol{W}_1^{(t)}, \boldsymbol{W}_2^{(t)}$ with $\{\boldsymbol{e}_{2,i}\}_{i\in\mathcal{B}^{(t)}}$ and $\{\boldsymbol{e}_{1,i}\}_{i\in\mathcal{B}^{(t)}}$, respectively ;
- Set $\boldsymbol{W}_1^{(t,0)} = \boldsymbol{W}_1^{(t)}$ and $\boldsymbol{W}_2^{(t,0)} = \boldsymbol{W}_2^{(t)}$;
- **for** $t' = 0, \ldots, T_u - 1$ **do** `// Multiple updates`
  - Compute mini-batch gradient of Equation (12) w.r.t. $\boldsymbol{W}_1^{(t,t')}, \boldsymbol{W}_2^{(t,t')}$;
  - Update $\boldsymbol{W}_1^{(t,t'+1)}, \boldsymbol{W}_2^{(t,t'+1)}$ with an optimizer ;
- Set $\boldsymbol{W}_1^{(t+1)} = \boldsymbol{W}_1^{(t,T_u)}$ and $\boldsymbol{W}_2^{(t+1)} = \boldsymbol{W}_2^{(t,T_u)}$;
- Compute $\alpha_1^{(t+1)}(\boldsymbol{x}_i)$ and $\alpha_2^{(t+1)}(\boldsymbol{z}_i)$ for $(\boldsymbol{x}_i, \boldsymbol{z}_i) \in \mathcal{B}^{(t)}$;
- Compute mini-batch gradient of Equation (12) w.r.t $\boldsymbol{w}^{(t)}, \tau^{(t)}$;
- Update $\boldsymbol{w}^{(t+1)}$ and $\tau^{(t+1)}$ with an optimizer;

Acceleration. We develop two techniques to accelerate training. First, we perform **multiple NPN updates** before updating the CLIP model. Since each update of the encoders changes the loss landscape of the NPNs, multiple updates enables the NPNs to maintain the same pace as the encoders and produce more accurate normalizers. In practice, we find that a small number of updates (e.g., $T_u = 10$) is sufficient. Since the NPNs are lightweight networks, the additional cost is minimal (c.f. Appendix B.4 for empirical results). Second, we apply **periodic re-initialization** of the NPNs by using randomly sampled text embeddings $\{\boldsymbol{e}_{2,i}\}$ to reset $\boldsymbol{W}_1$ and their corresponding image embeddings $\{\boldsymbol{e}_{1,i}\}$ to reset $\boldsymbol{W}_2$. This also helps mitigate the convergence gap between the CLIP model and the NPNs. This procedure is motivated by the observation that $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ act as compact summaries of all text and image embeddings. Together, these two techniques ensure that the NPNs remain well-aligned with the evolving encoders, leading to more effective training.

5 Experiments

Experiment Settings. In all the experiments, we train a CLIP model on an image-text dataset with a given compute budget (i.e., number of samples to be processed) using 8 NVIDIA H100 GPUs. The text encoder of the CLIP model is a Transformer (Vaswani et al., 2017), and the image encoder is either a ViT (Dosovitskiy et al., 2021) or a ResNet (He et al., 2016). We consider five datsets, including CC3M (Changpinyo et al., 2021), CC12M (Sharma et al., 2018) and three subsets of the DFN dataset at different scales (Fang et al., 2023a), ranging from 14M to 192M and 1B. The details of the experiment settings, including batch size and training budget for each dataset, can be found in Table 2. Ablation studies are conducted on the CC3M dataset and the DFN-14M dataset.

Evaluation Metrics. Throughout the section, we evaluate the performance of trained models on zero-shot classification and retrieval tasks. Specifically, we leverage the Datacomp benchmark (Gadre et al., 2023) and report the average performance on its 38 tasks (denoted as Datacomp Average). Moreover, we report the average performance on two subsets of the 38 tasks: (1) ImageNet & Variants, which consists of classification tasks on ImageNet-related datasets; (2) Retrieval, which consists of retrieval tasks. More information can be found in Appendix B.2.

Hyperparameters. For the NPNs, we set the number of columns $m = 4096$, restart frequency $T_r = 500$, and number of updates per iteration $T_u = 10$. We use the AdamW optimizer (Loshchilov & Hutter, 2019) to train the CLIP model and the AdaGrad optimizer (Duchi et al., 2011) to train the NPNs. We provide more details on other hyper-parameters in Appendix B.1.

5.1 Comparison with Baselines

In this subsection, we provide comparison between our proposed NeuCLIP and several strong baselines, including OpenCLIP (Cherti et al., 2023), FastCLIP (Wei et al., 2024), SigLIP (Zhai et al., 2023) and AmorLIP (Sun et al., 2025). For OpenCLIP and SigLIP, we use the implementation from

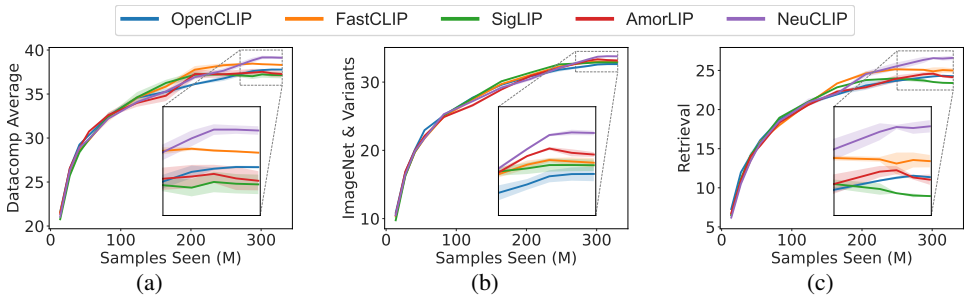


Figure 1: Performance curves of different methods on DFN-14M. (a): Datacomp Average performance. (b): ImageNet & Variants performance. (c): Retrieval performance.

Table 1: Datacomp Average performance of different methods trained on different datasets.

Method	CC3M	CC12M	DFN-14M	DFN-192M	DFN-1B
OpenCLIP	21.84	27.91	37.78	54.58	53.20
FastCLIP	24.74	31.50	38.45	54.72	53.57
SigLIP	22.19	28.60	37.23	54.26	53.22
AmorLIP	22.89	29.86	37.53	53.83	53.08
NeuCLIP	**25.08**	**31.89**	**39.16**	**54.90**	**53.74**

open_clip (Ilharco et al., 2021). For FastCLIP and AmorLIP, we use their released code, respectively. For experiments on CC3M, CC12M and DFN-14M, we repeat each method three times with different random seeds and report the mean. The Datacomp Average performance of different methods on the different datasets are presented in Table 1, and the full evaluation results are deferred to Appendix B.5. Additionally, we plot the performance curves during training in Figure 1.

The first observation we have is that NeuCLIP outperforms all other methods on all datasets, indicating the effectiveness of our approach. Secondly, from Figure 1 we find that NeuCLIP achieves larger improvement at the later stage of training. Note that in Algorithm 1, we optimize the NPNs $\boldsymbol{W}_1^{(t)}, \boldsymbol{W}_2^{(t)}$ given a fixed CLIP model $\boldsymbol{w}^{(t)}, \tau^{(t)}$. At later stage of training, the change in $\boldsymbol{w}^{(t)}, \tau^{(t)}$ becomes smaller, enabling the learning of NPNs to be more efficient for the updated encoders. Thirdly, for AmorLIP, we observe variation between our results and their reported results, since the datasets we used are not exactly the same as theirs (c.f. Appendix B.5). Finally, we note that all methods perform worse on DFN-1B than on DFN-192M because only 1B samples are processed for DFN-1B, compared to 1.3B for DFN-192M (cf. Table 2).

5.2 Ablation Study

In this subsection, we conduct ablation study of different components in NeuCLIP. We run all the experiments on DFN-14M or CC3M, where the setting is the same as in Table 2.

Comparison with AmorLIP's Design. The main differences between the design of AmorLIP and NeuCLIP lie in the training objective and model architecture of the NPN: (1) AmorLIP employs two separate objectives to train the CLIP model and the NPNs respectively, while we leverage a unified objective. Specifically, AmorLIP uses the following objective to train their NPNs:

$$\frac{1}{2|\mathcal{S}|}\sum\nolimits_{\boldsymbol{x}_i,\boldsymbol{z}_i\in\mathcal{S}}\left(\|\alpha_{1,i}-\log(\varepsilon+g_1(\boldsymbol{w},\tau;i,\mathcal{S}))\|^2+\|\alpha_{2,i}-\log(\varepsilon+g_2(\boldsymbol{w},\tau;i,\mathcal{S}))\|^2\right),$$

where $\alpha_{1,i}$ is the predicted log-normalizer for $\boldsymbol{x}_i$ and $\alpha_{2,i}$ is the predicted log-normalizer for $\boldsymbol{z}_i$. (2) AmorLIP chooses Multi-Layer Perceptrons (MLPs) as their NPNs, while in NeuCLIP we use single-layered NPNs that take advantage of the inductive bias in the optimal solutions of α. In order to provide a comparison between these design choices, we conduct the following experiments: (1) In Line 7 of Algorithm 1, we compute the gradient of the NPNs using the above equation, in which case the objectives for training the CLIP model and the NPNs are not unified anymore. (2) We instantiate the NPNs with MLPs, and initialize them with random weights, which follows the practice of Sun et al. (2025). We present the Datacomp Average performance of different objectives

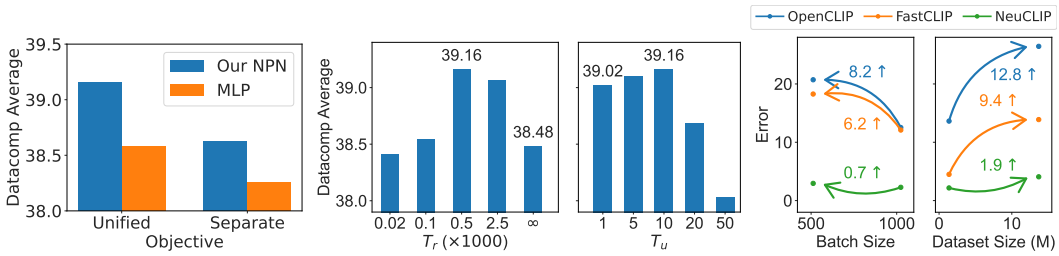


Figure 2: (a): Ablation study of training objective and architecture of NPNs. (b): Ablation study of restart frequency of NPNs (left) and number of updates (right). (c): Estimation error of NPNs.

and architectures in Figure 2a. From the results we can observe that learning with the unified objective yields better performance than learning with two separate objectives, and our inductive-biased NPN design outperforms MLPs. We also conduct the same experiments on CC3M and CC12M. The results, along with full results on DFN-14M, are presented in Tables 12 to 14, where we reach a similar conclusion.

Restart Frequency. To investigate the impact of the restart frequency T_r, we conduct experiments with different values of T_r, where $T_r = \infty$ means the NPNs are never re-initialized. We plot the Datacomp Average performance of different T_r in the left part of Figure 2b, and present the full evaluation results in Table 15 in Appendix B.6. From the results we can observe that when $T_r > 500$, the performance decreases. Also, $T_r = 500$ gives better performance than no re-initialization (i.e., $T_r = \infty$). This is probably because the NPNs lack behind the updated encoders, making their estimations less accurate for updated encoders. Moreover, when T_r is small, the NPNs are frequently set to the mini-batch features. In this case the output of the NPNs is close to the mini-batch estimators, which also leads to degraded performance.

Multiple NPN Updates. Another strategy to mitigate the gap of convergence speed between the NPNs and the encoders is to update the NPNs multiple (T_u) times before updating the encoders. Specifically, we use the same batch of data to compute the stochastic gradient w.r.t. the NPNs' parameters across multiple updates to avoid expensive forward passes of the CLIP model. We conduct experiments with different T_u and plot the results in the right part of Figure 2b. From the results we can see that as the number of updates increases, the performance first increases, and starts to decrease when $T_u > 10$. The decrease is expected since we are using the same batch of data to update the NPNs, which will overfit to the batch and provide inaccurate estimation for other samples.

Estimation Error of Normalizers. From Section 3 we know the estimation error of FastCLIP increases when the dataset size increases or when the batch size decreases. To demonstrate the effectiveness of our approach, we compare the estimation error of normalizers in OpenCLIP, FastCLIP, and NeuCLIP under the following two settings. Firstly, we run each method on CC3M using two batch sizes (512 and 1024). For each run, we select five checkpoints such that the corresponding checkpoints across batch size settings have seen the same number of samples. At each checkpoint, we compute the estimation error as the mean squared error between the logarithm of the predicted normalizers and the true normalizers, and then report the mean across checkpoints. As shown on the left part of Figure 2c, the error of NeuCLIP increases only marginally when the batch size decreases, whereas OpenCLIP and FastCLIP exhibit a much larger increase. Secondly, we run each method on two datasets of different sizes: a subset of DFN-14M ($n = 1.37$M) and full DFN-14M ($n = 13.7$M). On the right part of Figure 2c, we plot the average error of five checkpoints selected using the same procedure as above. The results show that NeuCLIP is only slightly affected by the increase in dataset size, in contrast to OpenCLIP and FastCLIP that suffer significant degradation.

6 Conclusion

In this paper, we have studied the problem of efficiently approximating the normalizers in the contrastive loss for training CLIP models. We proposed a novel objective that allows us to jointly learn CLIP encoders and compact networks that predict the log-normalizers of image and text data. We proposed an alternating optimization algorithm to learn the encoders and the compact networks efficiently. We conducted extensive experiments to demonstrate the effectiveness of our algorithm, and reveal insights on training of the network through ablation study.

REFERENCES

Ibrahim M Alabdulmohsin, Xiaohua Zhai, Alexander Kolesnikov, and Lucas Beyer. Getting vit in shape: Scaling laws for compute-optimal model design. *Advances in Neural Information Processing Systems*, 36:16406–16425, 2023.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/97af07a14cacba681feacf3012730892-Paper.pdf.

Ido Ben-Shaul, Ravid Shwartz-Ziv, Tomer Galanti, Shai Dekel, and Yann LeCun. Reverse engineering self-supervised learning, 2023. URL https://arxiv.org/abs/2305.15614.

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021.

Jieneng Chen, Qihang Yu, Xiaohui Shen, Alan Yuille, and Liang-Chieh Chen. Vitamin: Designing scalable vision models in the vision-language era. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12954–12966, June 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. URL https://arxiv.org/abs/2002.05709.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2829, June 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL http://jmlr.org/papers/v12/duchi11a.html.

Talfan Evans, Shreya Pathak, Hamza Merzic, Jonathan Schwarz, Ryutaro Tanno, and Olivier J. Hénaff. Bad students make great teachers: Active learning accelerates large-scale visual understanding. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 264–280, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72643-9.

Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander Toshev, and Vaishaal Shankar. Data filtering networks. *arXiv preprint arXiv:2309.17425*, 2023a.

Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19358–19369, June 2023b.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, Eyal Orgad, Rahim Entezari, Giannis Daras, Sarah Pratt, Vivek Ramanujan, Yonatan Bitton, Kalyani Marathe, Stephen Mussmann, Richard Vencu, Mehdi Cherti, Ranjay Krishna, Pang Wei W Koh, Olga Saukh, Alexander J Ratner, Shuran Song, Hannaneh Hajishirzi, Ali Farhadi, Romain Beaumont, Sewoong Oh, Alex Dimakis, Jenia Jitsev, Yair Carmon, Vaishaal Shankar, and Ludwig Schmidt. Datacomp: In search of the next generation of multimodal datasets. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 27092–27112. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/56332d41d55ad7ad8024aac625881be7-Paper-Datasets_and_Benchmarks.pdf.

Luigi Grippof and Marco Sciandrone. Globally convergent block-coordinate techniques for unconstrained optimization. *Optimization Methods and Software*, 10(4):587–637, 1999. doi: 10.1080/10556789908805730. URL https://doi.org/10.1080/10556789908805730.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8340–8349, October 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15262–15271, June 2021b.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

Minsu Kim, Joohwan Ko, Taeyoung Yun, Dinghuai Zhang, Ling Pan, Woo Chang Kim, Jinkyoo Park, Emmanuel Bengio, and Yoshua Bengio. Learning to scale logits for temperature-conditional GFlowNets. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 24248–24270. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/kim24s.html.

Guanghui. Lan. *First-order and Stochastic Optimization Methods for Machine Learning*. Springer Series in the Data Sciences. Springer International Publishing, Cham, 1st ed. 2020. edition, 2020. ISBN 3-030-39568-5.

Xianhang Li, Zeyu Wang, and Cihang Xie. An inverse scaling law for clip training. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 49068–49087. Curran Associates, Inc., 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/996e2b446391fcb8bf32a3d1645cc799-Paper-Conference.pdf.

Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. Scaling language-image pre-training via masking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23390–23400, June 2023b.

Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Comput.*, 19(10):2756–2779, October 2007. ISSN 0899-7667. doi: 10.1162/neco.2007.19.10.2756. URL https://doi.org/10.1162/neco.2007.19.10.2756.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Qi Qian and Juhua Hu. Online zero-shot classification with clip. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 462–477, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-72980-5.

Zi-Hao Qiu, Quanqi Hu, Zhuoning Yuan, Denny Zhou, Lijun Zhang, and Tianbao Yang. Not all semantics are created equal: Contrastive self-supervised learning with automatic temperature individualization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28389–28421. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/qiu23a.html.

Zi-Hao Qiu, Siqi Guo, Mao Xu, Tuo Zhao, Lijun Zhang, and Tianbao Yang. To cool or not to cool? temperature network meets large foundation models via DRO. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=YWuSLBkfOw.

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. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5389–5400. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/recht19a.html.

R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

R.T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 1997. ISBN 9780691015866. URL https://books.google.com/books?id=1TiOka9bx3sC.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*, 2018.

Maohao Shen, Subhro Das, Kristjan Greenewald, Prasanna Sattigeri, Gregory W. Wornell, and Soumya Ghosh. Thermometer: Towards universal calibration for large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 44687–44711. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/shen24c.html.

Junhao Su, Changpeng Cai, Feiyu Zhu, Chenghao He, Xiaojie Xu, Dongzhi Guan, and Chenyang Si. Momentum auxiliary network for supervised local learning. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 276–292, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72661-3.

Haotian Sun, Yitong Li, Yuchen Zhuang, Niao He, Hanjun Dai, and Bo Dai. Amorlip: Efficient language-image pretraining via amortization. *arXiv preprint arXiv:2505.18983*, 2025.

Pavan Kumar Anasosalu Vasu, Hadi Pouransari, Fartash Faghri, Raviteja Vemulapalli, and Oncel Tuzel. Mobileclip: Fast image-text models through multi-modal reinforced training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15963–15974, June 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Bokun Wang, Yunwen Lei, Yiming Ying, and Tianbao Yang. On discriminative probabilistic modeling for self-supervised representation learning. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=s15HrqCqbr.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3eefceb8087e964f89c2d59e8a249915-Paper.pdf.

Xiao Wang, Ibrahim Alabdulmohsin, Daniel Salz, Zhe Li, Keran Rong, and Xiaohua Zhai. Scaling pre-training to one hundred billion data for vision language models. *arXiv preprint arXiv:2502.07617*, 2025b.

Yiping Wang, Yifang Chen, Wendan Yan, Alex Fang, Wenjing Zhou, Kevin Jamieson, and Simon Shaolei Du. Cliploss and norm-based data selection methods for multimodal contrastive learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 15028–15069. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/1b33deb9e1e7c31f05300b1c2d4a4f7d-Paper-Conference.pdf.

Xiyuan Wei, Fanjiang Ye, Ori Yonay, Xingyu Chen, Baixi Sun, Dingwen Tao, and Tianbao Yang. Fastclip: A suite of optimization techniques to accelerate clip training with limited resources. *arXiv preprint arXiv:2407.01445*, 2024.

Xiyuan Wei, Ming Lin, Fanjiang Ye, Fengguang Song, Liangliang Cao, My T. Thai, and Tianbao Yang. Model steering: Learning with a reference model improves generalization bounds and scaling laws. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=QC4dfobOLQ.

Hu Xu, Saining Xie, Xiaoqing Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying CLIP data. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5BCFlnfE1g.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 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, 2014.

Zhuoning Yuan, Yuexin Wu, Zi-Hao Qiu, Xianzhi Du, Lijun Zhang, Denny Zhou, and Tianbao Yang. Provable stochastic optimization for global contrastive learning: Small batch does not harm performance. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25760–25782. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/yuan22b.html.

Zhixiong Zeng and Wenji Mao. A comprehensive empirical study of vision-language pre-trained model for supervised cross-modal retrieval. *arXiv preprint arXiv:2201.02772*, 2022.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.

A Technical Details

A.1 Derivation of our New Objective and the Optimal Solution

When $f(\cdot) = -\log(\cdot)$, we have

$$f^*(y) = \max_x y \cdot x - f(x) = \max_x y \cdot x + \log(x).$$

From the first-order optimality condition we know $y + 1/x^* = 0$, which is $x^* = -1/y$. Substituting the optimal solution of x into the definition of $f^*(y)$, we get $f^*(y) = -\log(-y) - 1$. Since $-\log(\cdot)$ is a convex function, we have

$$f(x) = f^{**}(x) = \max_y x \cdot y - f^*(y) = \max_y x \cdot y + \log(-y) + 1.$$

Plugging in $x = \varepsilon + g_1(\boldsymbol{w}, \tau, i, \mathcal{S})$ into the above equation, we get

$$\begin{aligned} F(\boldsymbol{w}, \tau; \boldsymbol{x_i}) &= \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) \\ &= -1 \cdot (-\log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S}))) \\ &= -\max_y \{y \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \log(-y) + 1\} \\ &= \min_y \{-y \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) - \log(-y) - 1\} \\ &= \min_\alpha \{\exp(-\alpha) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + \alpha - 1\}, \end{aligned}$$

where the last equality uses a change of variable $\alpha = -\log(-y)$. This completes the derivation of the new objective. Moreover, to derive the optimal solution of α, we can leverage the first-order optimality:

$$-\exp(-\alpha^*) \cdot (\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S})) + 1 = 0,$$

which gives $\alpha^* = \log(\varepsilon + g_1(\boldsymbol{w}, \tau; i, \mathcal{S}))$.

Algorithm 2: The FastCLIP Algorithm

Input: Model $\boldsymbol{w}^{(0)}$, temperature $\tau^{(0)}$, estimators $\boldsymbol{u}_1, \boldsymbol{u}_2$, dataset $\mathcal{S}$, number of iterations T

for $t = 0, \ldots, T-1$ **do**

Randomly sample a mini-batch $\mathcal{B}^{(t)} \subset \mathcal{S}$;

Update $u_{1,i}^{(t+1)}$ and $u_{2,i}^{(t+1)}$ using Equation (3) for $i \in \mathcal{B}^{(t)}$;

Set $u_{1,i}^{(t+1)} = u_{1,i}^{(t)}$ and $u_{2,i}^{(t+1)} = u_{2,i}^{(t)}$ for $i \notin \mathcal{B}^{(t)}$;

Compute gradient estimators for $\boldsymbol{w}^{(t)}, \tau^{(t)}$ using Equations (4) and (13), respectively

$$\begin{aligned} &\frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \log(u_{1,i}^{(t+1)}) + \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \log(u_{2,i}^{(t+1)}) + 2\rho \\ &+\tau^t \cdot \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \frac{1}{u_{1,i}^{(t+1)}} \cdot \nabla_\tau g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}) \\ &+\tau^{(t)} \cdot \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \frac{1}{u_{2,i}^{(t+1)}} \cdot \nabla_\tau g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}). \end{aligned} \tag{13}$$

Update $\boldsymbol{w}^{(t+1)}$ and $\tau^{(t+1)}$ using an optimizer;

A.2 Inducing the u Update of SogCLR from Equation (6)

Indeed, the u sequence update in Equation (3) and the gradient estimator of $\boldsymbol{w}$ in Equation (4) can be derived for optimizing Equation (6). To illustrate this, we consider a stochastic block mirror descent update of $\bar{y}_{1,i} = \exp(-\alpha_{1,i})$ and $\bar{y}_{2,i} = \exp(-\alpha_{2,i})$ for $(\boldsymbol{x_i}, \boldsymbol{z_i}) \in \mathcal{B}^t$ at the t-th iteration :

$$\bar{y}_{1,i}^{(t+1)} = \arg\min_{y_{1,i}} \; y_{1,i} \cdot (\varepsilon + g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)})) - \log(y_{1,i}) + \frac{1}{\eta} \cdot D(y_{1,i}, \bar{y}_{1,i}^{(t)}),$$

$$\bar{y}_{2,i}^{(t+1)} = \arg\min_{y_{2,i}} \; y_{2,i} \cdot (\varepsilon + g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)})) - \log(y_{2,i}) + \frac{1}{\eta} \cdot D(y_{2,i}, \bar{y}_{2,i}^{(t)}),$$

where $D(u, v) = -\log(u) + \log(v) + \frac{1}{v}(u - v)$ is the Bregman divergence induced by $-\log(\cdot)$ (also known as the Itakura–Saito distance), and $\eta > 0$ is a hyperparameter. We can derive closed-form updates of $\bar{y}_{1,i}^{(t+1)}, \bar{y}_{2,i}^{(t+1)}$ as follows. By the first-order optimality condition, we have

$$\left(\varepsilon + g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}) - \frac{1}{\bar{y}_{1,i}^{(t+1)}}\right) + \frac{1}{\eta}\left(-\frac{1}{\bar{y}_{1,i}^{(t+1)}} + \frac{1}{\bar{y}_{1,i}^{(t)}}\right) = 0.$$

Rearranging the terms, we get

$$\frac{1}{\bar{y}_{1,i}^{(t+1)}} = \frac{1}{1+\eta} \cdot \frac{1}{\bar{y}_{1,i}^{(t)}} + \frac{\eta}{1+\eta} \cdot \left(\varepsilon + g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)})\right),$$

which can be mapped to Equation (3) with $u_{1,i}^{(t)} = 1/\bar{y}_{1,i}^{(t)}$ and $\gamma = \eta/(1+\eta)$. With $\bar{y}_{1,i}^{(t+1)}, \bar{y}_{2,i}^{(t+1)}$, the stochastic gradient of Equation (6) w.r.t. $\boldsymbol{w}^{(t)}$ is given by

$$\tau^{(t)} \cdot \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \bar{y}_{1,i}^{(t+1)} \cdot \nabla_{\boldsymbol{w}} g_1(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}) + \tau^{(t)} \cdot \frac{1}{|\mathcal{B}^{(t)}|} \sum_{i \in \mathcal{B}^{(t)}} \bar{y}_{2,i}^{(t+1)} \cdot \nabla_{\boldsymbol{w}} g_2(\boldsymbol{w}^{(t)}, \tau^{(t)}; i, \mathcal{B}^{(t)}),$$

which is exactly same as in Equation (4) with $u_{1,i}^{(t)} = 1/\bar{y}_{1,i}^{(t)}$.

A.3 Details of the FastCLIP Algorithm

The FastCLIP algorithm (Wei et al., 2024) is presented in Algorithm 2.

B Additional Experiment Results

B.1 Details of Experiment Settings

We present the details of the experiment settings in Table 2.

Table 2: Details of experiment settings. "Size" denotes the number of image-text pairs we successfully downloaded. "Samples" denotes the number of samples that are processed for training, which equals the total number of iterations times global batch size. "Batch Size" denotes the global batch size.

Dataset	Dataset Size	Samples Seen	Vision Encoder	Batch Size
CC3M	2.7M	100M	ResNet-50	1024
CC12M	9.2M	300M	ViT-B/32	2048
DFN-14M	13.7M	320M	ViT-B/32	4096
DFN-192M	192M	1.3B	ViT-B/16	5120
DFN-1B	1.0B	1.0B	ViT-B/16	5120

Table 3: Hyperparameters for training the CLIP model.

Dataset	lr	lr of τ	wd	warmup	ρ
CC3M	1e-3	1.25e-4	0.1	10000	6.5
CC12M	4e-4	5e-5	0.1	10000	8.5
DFN-14M	5e-4	6.25e-5	0.2	500	11.0
DFN-192M	3.125e-4	3.9e-5	0.2	500	11.0
DFN-1B	3.125e-4	3.9e-5	0.2	500	11.0

Hyperparameters. In Table 3, we present more hyperparameters for training the CLIP model. For each dataset, we tune the learning rate between 1e-4 and 1e-3, and tune the learning rate of the temperature parameter between 1/8 of the learning rate and the learning rate. We set the weight decay (wd) following previous works (Gadre et al., 2023; Wei et al., 2024). For CC3M and CC12M, we set the value of ρ following Wei et al. (2024). For the DFN datasets, we tune ρ between 8.0 and 13.0. In Table 4, we present hyperparameters for training the NPNs. We tune the learning rate between 1e-4 and 100.0 and the weight decay between 0.0 and 1.0.

B.2 Details of the Datacomp Benchmark

The Datacomp Benchmark (Gadre et al., 2023) consists of 38 tasks in total, including 35 zero-shot image classification tasks and 3 zero-shot retrieval tasks. The performance metric for classification tasks is top-1 accuracy, and the performance metric for retrieval tasks is the average of image recall at 1 and text recall at 1. The "ImageNet & Variants" subset consists of ImageNet-1K (Deng et al., 2009) and 6 distribution shift datasets (Wang et al., 2019; Recht et al., 2019; Hendrycks et al., 2021a;b; Barbu et al., 2019), and the "Retrieval" subset consists of Flickr30K (Young et al., 2014) and MSCOCO (Chen et al., 2015).

B.3 Simultaneous vs. Alternating Optimization

In this subsection, we provide comparison for two approaches solving Equation (12). The first approach is a simple gradient-based algorithm that treats all parameters as a whole and updates them simultaneously, which is presented in Algorithm 3. The second one is our NeuCLIP algorithm (Algorithm 1), which optimizes the CLIP model and the NPNs in an alternating manner. We conduct experiments on CC3M and plot the performance of the two approaches in Figure 3. From the results we can see that the joint optimization algorithm performs much worse than the alternating optimization algorithm.

Table 4: Hyperparameters for training the NPNs.

Optimizer	lr	wd	m	T_r	T_u
AdaGrad	1.0	0.0	4096	500	10

Algorithm 3: Simple (Stochastic) Gradient-based Algorithm for Solving Equation (12)

Input: Model $\boldsymbol{w}^0$, Temperature τ^0, Prototypes $\boldsymbol{W}_1^0, \boldsymbol{W}_2^0$, Dataset $\mathcal{S}$, Number of iterations T, Restart frequency T_r, Number of updates of networks per iteration T_u

for $t = 0, \ldots, T-1$ **do**
- Randomly sample a mini-batch $\mathcal{B}^{(t)} \subset \mathcal{S}$;
- Compute mini-batch gradient of (12) w.r.t. $\boldsymbol{w}^{(t)}, \tau^{(t)}, \boldsymbol{W}_1^{(t)}, \boldsymbol{W}_2^{(t)}$;
- Update $\boldsymbol{w}^{(t+1)}, \tau^{(t+1)}, \boldsymbol{W}_1^{(t+1)}, \boldsymbol{W}_2^{(t+1)}$ with AdamW;

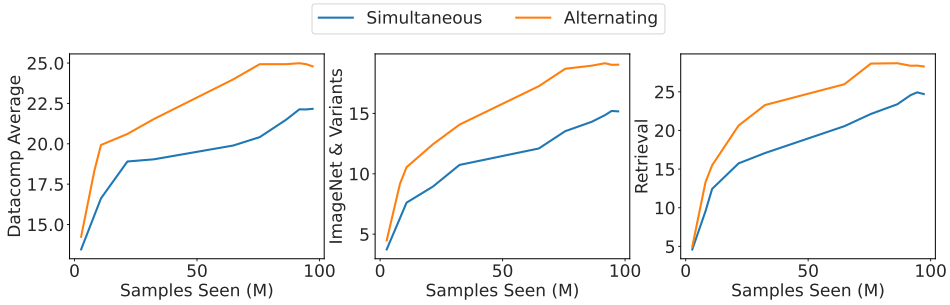


Figure 3: Comparison between simultaneous and alternating optimization.

B.4 Training Cost of the Normalizer-Prediction Network

In this subsection, we investigate the additional training cost incurred by the NPNs. We profile the running time of NeuCLIP on CC3M, DFN-14M and DFN-192M (where the vision encoder is ResNet50, ViT-B/32 and ViT-B16 respectively) using the PyTorch (Paszke et al., 2019) Profiler. We present the results in Table 5, from which we can observe that the additional cost of the NPNs remains low.

B.5 Comparison with Baselines

In this subsection, we provide more experiment results of different methods on various datasets. In Tables 6 to 10, we present the evaluation results of different methods trained on CC3M to DFN-1B.

Variation in Performance of AmorLIP. On CC3M and CC12M, we find that the results of our reproduction of AmorLIP is different from those reported in the AmorLIP paper. For example, their reported results of FastCLIP are lower than ours while that of AmorLIP are higher (c.f. Table 11). We would like to note that the training datasets used by Sun et al. (2025) and us are in fact different. This is because the two datasets are distributed with their metadata only, i.e., texts and links to their corresponding images. During the downloading process, not all images would be successfully downloaded. Specifically, Sun et al. (2025) reported a size of 2,274,566 samples for CC3M and 8,059,642 samples for CC12M, while our CC3M and CC12M datasets contain 2,723,840 and 9,187,328 samples, respectively. We tune the parameters of AmorLIP and still could not achieve the reported performance, thus we believe the difference in performance is due to variation in datasets. On the other hand, if we compare their reported results with the results of NeuCLIP, we still observe an improvement of NeuCLIP over AmorLIP.

Table 5: Running time of NeuCLIP on different datasets. NPN Time denotes the training time of the NPNs in one iteration. Total Time denotes the running time of one iteration. Percentage is equal to NPN Time / Total Time.

Dataset	Vision Encoder	NPN Time (ms)	Total Time (ms)	Percentage
CC3M	ResNet50	49.23 ± 1.10	529.09 ± 21.79	9.30%
DFN-14M	ViT-B/32	54.17 ± 3.98	897.79 ± 31.89	6.03%
DFN-192M	ViT-B/16	56.43 ± 2.83	944.41 ± 29.40	5.98%

Table 6: Evaluation results of different methods trained on CC3M. We run each method for 3 times with different random seeds, and report the average performance over 3 training runs along with standard deviation.

Method	Datacomp Average	ImageNet & Variants	Retrieval
OpenCLIP	21.84 ± 0.23	14.73 ± 0.22	22.25 ± 0.38
FastCLIP	24.74 ± 0.35	19.09 ± 0.20	29.56 ± 0.25
SigLIP	22.19 ± 0.11	16.08 ± 0.16	22.13 ± 0.52
AmorLIP	22.89 ± 0.20	17.78 ± 0.43	24.32 ± 0.53
NeuCLIP	**25.08** ± 0.39	**19.85** ± 0.37	**30.53** ± 0.37

Table 7: Evaluation results of different methods trained on CC12M. We run each method for 3 times with different random seeds, and report the average performance over 3 training runs along with standard deviation.

Method	Datacomp Average	ImageNet & Variants	Retrieval
OpenCLIP	27.91 ± 0.77	21.21 ± 0.04	26.94 ± 0.59
FastCLIP	31.50 ± 0.43	24.61 ± 0.05	32.33 ± 0.48
SigLIP	28.60 ± 0.35	22.67 ± 0.05	27.99 ± 0.25
AmorLIP	29.86 ± 0.83	23.48 ± 0.62	28.97 ± 0.77
NeuCLIP	**31.89** ± 0.15	**25.09** ± 0.12	**32.93** ± 0.16

Table 8: Evaluation results of different methods trained on DFN-14M. We run each method for 3 times with different random seeds, and report the average performance over 3 training runs along with standard deviation.

Method	Datacomp Average	ImageNet & Variants	Retrieval
OpenCLIP	37.78 ± 0.05	32.65 ± 0.20	24.33 ± 0.06
FastCLIP	38.45 ± 0.06	33.03 ± 0.06	25.15 ± 0.11
SigLIP	37.23 ± 0.10	32.89 ± 0.17	23.97 ± 0.20
AmorLIP	37.53 ± 0.33	33.35 ± 0.04	24.58 ± 0.19
NeuCLIP	**39.16** ± 0.20	**33.79** ± 0.07	**26.60** ± 0.28

Table 9: Evaluation results of different methods trained on DFN-192M.

Method	Datacomp Average	ImageNet & Variants	Retrieval
OpenCLIP	54.58	55.16	50.42
FastCLIP	54.72	55.44	50.53
SigLIP	54.26	55.09	50.49
AmorLIP	53.83	55.09	**51.43**
NeuCLIP	**54.90**	**55.88**	51.39

Table 10: Evaluation results of different methods trained on DFN-1B.

Method	Datacomp Average	ImageNet & Variants	Retrieval
OpenCLIP	53.20	54.91	50.17
FastCLIP	53.57	55.15	50.56
SigLIP	53.22	55.06	50.46
AmorLIP	53.08	55.12	50.52
NeuCLIP	**53.74**	**55.52**	**50.96**

Table 11: Datacomp Average performance of FastCLIP and AmorLIP from Sun et al. (2025) and our reproduction.

Method	Source	CC3M	CC12M
FastCLIP	Sun et al. (2025)	23.46	29.00
FastCLIP	Ours	25.08	31.89
AmorLIP	Sun et al. (2025)	24.11	30.66
AmorLIP	Ours	22.89	29.86

Table 12: Ablation of training objective and model architecture on CC3M.

Objective	Architecture	Datacomp Average	ImageNet & Variants	Retrieval
Unified	Our NPN	**25.08**	**19.85**	**30.53**
Unified	MLP	24.84	19.28	29.50
Separate	Our NPN	24.19	19.20	29.38
Separate	MLP	24.02	19.09	29.08

B.6 Ablation Study

In this subsection, we present detailed results of the ablation study. In Tables 12 to 14, we present evaluation results of different objectives and architectures on CC3M, CC12M and DFN-14M, respectively. In Tables 15 and 16, we present the ablation results of the restart frequency T_r and the number of updates T_u, respectively. In Figure 4, we plot the estimation error of different methods at different number of samples seen.

Table 14: Ablation of training objective and model architecture on DFN-14M.

Objective	Architecture	Datacomp Average	ImageNet & Variants	Retrieval
Unified	Ours NPN	**39.16**	**33.79**	**26.60**
Unified	MLP	38.58	33.19	26.35
Separate	Ours NPN	38.63	33.70	25.98
Separate	MLP	38.26	33.12	25.86

Table 15: Ablation of restart frequency T_r.

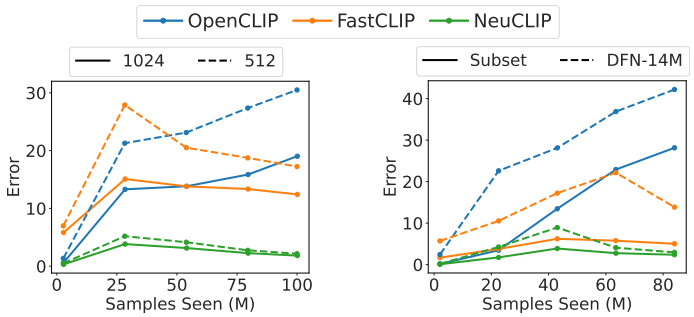
T_r	Datacomp Average	ImageNet & Variants	Retrieval
0	38.48	33.18	25.79
20	38.41	33.14	26.30
100	38.54	33.37	26.05
500	**39.16**	**33.79**	**26.60**
2500	39.06	33.29	26.25
12500	39.07	33.21	26.18

Table 13: Ablation of training objective and model architecture on CC12M.

Objective	Architecture	Datacomp Average	ImageNet & Variants	Retrieval
Unified	Our NPN	**31.89**	**25.09**	**32.93**
Unified	MLP	31.43	25.04	32.30
Separate	Our NPN	31.05	25.04	31.34
Separate	MLP	30.94	24.77	31.12

Table 16: Ablation of number of updates T_u.

T_u	Datacomp Average	ImageNet & Variants	Retrieval
1	39.02	33.50	**26.65**
5	39.10	33.60	26.48
10	**39.16**	**33.79**	26.60
20	38.68	33.57	26.05
50	38.03	33.35	26.46



(a) (b)

Figure 4: (a): Estimation error of different methods with batch size 1024 (solid lines) or 512 (dashed lines). (b): Estimation error of different methods on subset of DFN-14M (solid lines) or DFN-14M (dashed lines).

C The Use of Large Language Models (LLMs)

We use LLMs to help find recent applications of CLIP models, and to help search for works that leverage auxiliary networks in other fields than CLIP training, such as Computer Vision and Natural Language Processing. We also use LLMs to help polish up writing.