# Contrastive Vision-Language Pre-training with Limited Resources

Quan Cui[1,2], Boyan Zhou[1], Yu Guo[1], Weidong Yin[1], Hao Wu[1⋆], Osamu Yoshie[2], and Yubo Chen[1]

[1] ByteDance
[2] Waseda University
cui-quan@toki.waseda.jp, wuhao.5688@bytedance.com

**Abstract.** Pioneering dual-encoder pre-training works (*e.g.*, CLIP and ALIGN) have revealed the potential of aligning multi-modal representations with contrastive learning. However, these works require a tremendous amount of data and computational resources (*e.g.*, billion-level web data and hundreds of GPUs), which prevent researchers with limited resources from reproduction and further exploration. To this end, we propose a stack of novel methods, which significantly cut down the heavy resource dependency and allow us to conduct dual-encoder multi-modal representation alignment with limited resources. Besides, we provide a reproducible baseline of competitive results, namely ZeroVL, with only 14M publicly accessible academic datasets and 8 V100 GPUs. Additionally, we collect 100M web data for pre-training, and achieve comparable or superior results than state-of-the-art methods, further proving the effectiveness of our methods on large-scale data. We hope that this work will provide useful data points and experience for future research in contrastive vision-language pre-training. Code is available at https://github.com/zerovl/ZeroVL.

**Keywords:** Multi-Modal Representation Learning, Contrastive Learning, Language-Image Pre-training, Limited Resources

## 1 Introduction

Large-scale representation pre-training has become the de-facto approach in vision [5,6,17,45], language [10,30,18] and vision-language [35,22] modeling tasks. In the vision-language pre-training field, most mainstream approaches fall into one of two classes: single-encoder [40,28,37,7,29,51,13,20,23,27] and dual-encoder [35,22]. Typical single-encoder approaches focus on learning semantic alignments between image regions and text entities with a single backbone network, greatly benefiting various downstream multi-modal tasks, *e.g.*, VQA [1,50,14], VCR [48] and NLVR [38,39], *etc.*. In real-scenario applications [34], dual-encoder pre-training approach could be preferable for its flexibility. For one thing, downstream tasks of either modality can benefit from the pre-training. For another, dual-encoder approaches are more efficient than single-encoder approaches on popular multi-modal industrial applications, *e.g.*, cross-modal matching and retrieval tasks [4,21].

⋆ Corresponding author.

| method | **computation** | | **data** | MS-COCO | | F30K | |
|---|---|---|---|---|---|---|---|
| | device | count | | zs. | ft. | zs. | ft. |
| CLIP [35] | V100 | 256 | 400M | 400.2 | – | 540.6 | – |
| ALIGN [22] | $TPU_{v3}$ | 1,024 | 1800M | 425.3 | 500.4 | **553.3** | **576.0** |
| baseline | V100 | 8 | 14.2M | 371.6 | 471.9 | 483.3 | 553.0 |
| ZeroVL | V100 | 8 | 14.2M | 425.0 | 485.0 | 536.2 | 561.6 |
| ZeroVL† | V100 | 8 | 100M | **442.1** | **500.5** | 546.5 | 573.6 |

**Table 1.** Statistics of training resources and cross-modal retrieval RSUM scores [4,43,3]. "zs." and "ft." represent zero-shot and fine-tuned settings. "†" means pre-training with 100M web data.

Recent works [35,22] have demonstrated that, by aligning visual and language representations with the contrastive loss, a simple dual-encoder architecture is able to yield state-of-the-art representation learning performances. However, we notice a significant problem which might obstruct the progress in this research direction, *i.e.*, pioneering works require a tremendous amount of vision-linguistic corpus and computational resources for training, and such heavy resource dependency prevents researchers with limited resources from reproduction and further explorations. For instance, CLIP [35] and ALIGN [22] respectively collected 400M and 1.8B web image-text pairs and trained models with 256 V100 GPUs and 1,024 TPU cores. Such experimental environments present a big challenge for the most researchers, and further lead to a lack of commonly reproducible benchmarks for dual-encoder model, making it hard to validate novel methods.

To alleviate the problems above, we design a comprehensive training pipeline with only open-source academic datasets and limited computational resources. Specifically, we propose a collection of novel methods to deal with limited data and computation, respectively. Our proposed methods boost model performances while only introducing marginal overhead to both computation and implementation. As shown in Table 1, we achieve competitive results with ∼14M academic data and 8 V100 GPUs, greatly alleviating the heavy dependency on data and computation of contrastive language-image pre-training. To further demonstrate the effectiveness of our method on large-scale data, we collect 100M web image-text images and conduct pre-training without fine-tuning hyper-parameters. Surprisingly, our method successfully outperforms CLIP and achieves comparable results with ALIGN on pre-training and fine-tuning tasks.

## 2 A Naive Baseline

In this section, we build up a naive baseline for stacking our methods and polishing it to a strong one. Methods are related to *training with limited data* and *training with limited computation resource*, which will be discussed in Sec. 3 and 4.

### 2.1 Pre-Training Datasets

To ensure reproducibility, only publicly accessible academic datasets are leveraged to demonstrate the effectiveness of our methods. The statistics of collected image-text pair datasets are reported in Table 2. Four widely-used image-text pair

| | Pre-training | | | | | Test | |
|---|---|---|---|---|---|---|---|
| | Total | SBU | VG | CC3M | CC12M | MS-COCO | F30K |
| #image | 14.23M | 0.86M | 0.50M | 2.81M | 10.06M | 5K | 1K |
| #text | 14.23M | 0.86M | 0.50M | 2.81M | 10.06M | 25K | 5K |

**Table 2.** The statistics of datasets for pre-training and test.

datasets are selected for pre-training, *i.e.*, *(1) SBU Captioned Photos (SBU)* [33], *(2) Visual Genome (VG)* [25], *(3) Conceptual Captions 3M (CC3M)* [36], and *(4) Conceptual 12M (CC12M)* [2] datasets. Detailed introductions are attached in the appendix.

### 2.2 Baseline Settings

Baseline settings are elaborated from the data, model, and training perspectives.

**Data preparation.** Batches are comprised by randomly sampling image-text pairs from pre-training datasets. Following [35,22], each image is randomly cropped to a rectangular region with aspect ratio sampled in $[3/4, 4/3]$ and area sampled in $[60\%, 100\%]$, then resized to 224×224 resolution. Regarding the corresponding text, we use a percentage of 20% input words for processing. For each word, we mask it, replace it with a random word, or delete it with a probability of 50%, 10% and 40%, respectively. During test, images are resized to 256×256 and center cropped to 224×224, while no specific process is applied to texts.

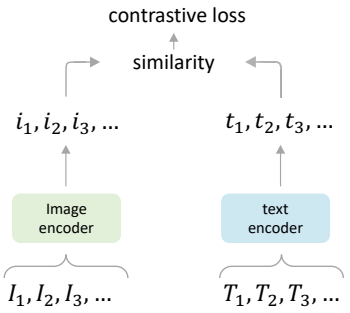


**Fig. 1.** Illustration of the dual-encoder model architecture.

**Model architecture.** Inspired by [35,22], we employ a simple dual-encoder model to align visual and language representations of image-text pairs via a contrastive loss. The framework is illustrated in Figure 1. Image and text encoders are ViT-B/16 [11] and BERT-Base [10], respectively. `[CLS]` tokens from image and text encoders are extracted and then projected to compact embeddings for calculating the contrastive loss.

**Training.** AdamW [24,31] optimizer is used for training and the weight decay is 1e-3. The dual-encoder model is trained for 20 epochs on 8 Nvidia V100 GPUs with a batch size of 1,024. The learning rate is initialized to 1e-4 and follows a cosine decay schedule. Notably, we set a minimum learning rate 1e-5 to avoid over-fitting. The embedding dimension for image and text representations is 512 and the trainable temperature of contrastive loss is initialized to 0.02.

### 2.3 Evaluations

**Metrics.** Typically, multi-modal retrieval tasks are assessed with the recall at K (R@K) metric, with K = $\{1, 5, 10\}$. We follow [43,3,4] to use RSUM as the metric to reveal the overall performance, which is defined as the sum of recall metrics at K = $\{1, 5, 10\}$ of both image-to-text and text-to-image retrieval tasks.
**Test datasets.** Following the standard practice in [35,22,4,12,43,3], we evaluate representations of pre-trained models by carrying out *zero-shot* image-text retrieval tasks on test sets of *(1) MS-COCO Captions Karpathy's split (MS-COCO)* and *(2) Flickr 30K (F30K)* datasets. MS-COCO and F30K results are reported with 5K and 1K test images, respectively.

## 3 Training with Limited Data Resource

Due to the copyright or technical issues, publicly accessible image-text academic datasets are greatly limited. The common practice to construct vision-linguistic corpus is collecting datasets from multiple sources. However, it brings in the dataset bias issue, which is caused by different collection manners of these datasets. Besides, limited data could suffer from the over-fitting problem, and seldom efforts were made for creating extra data for multi-modal pre-training. In this section, we study how to take full advantages of limited data from these two perspectives, *i.e.*, (1) leveraging biased data and (2) creating extra data.

### 3.1 Leveraging Biased Data with Debiased Sampling

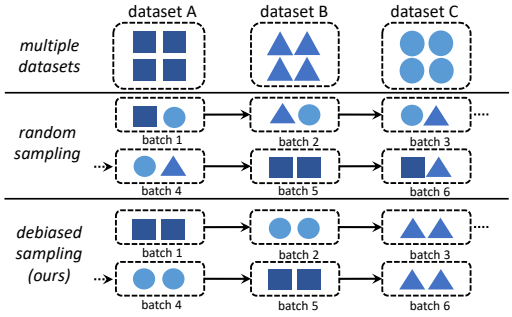


**Fig. 2.** Illustration of sampling strategies.

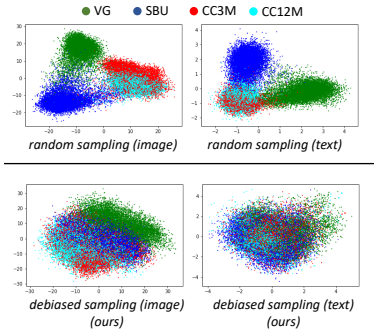


**Fig. 3.** Illustration of image and text embeddings.

**Random sampling brings in dataset bias.** Random sampling is an intuitive and widely used strategy, which randomly constructs training batches with all available data, as illustrated in Figure 2. However, when a batch is composed of

samples from different datasets, models could be driven to distinguish negative samples by hacking the source information, *i.e.*, learning the dataset bias. For instance, dataset A is mainly composed of *natural scenery photos with long captions*, while dataset B is mainly comprised of *people with short captions*. To distinguish samples from A and B, models are allowed to remember the dataset bias on image contents and caption lengths. To prove this, we first carry out visualizations to show the biased distribution of representations learned by random sampling. Then, we delve into the gradient of InfoNCE loss and provide evidences that data bias influences the model optimization.

**Dataset bias leads to biased representation distributions.** In the upper part of Figure 3, we visualize image and text embeddings learned with random sampling. Intra-dataset representations are closely gathered, while inter-dataset representations are separated. Representations are separated to three parts, *i.e.*, VG, SBU and "CC3M+CC12M". Since CC3M and CC12M are composed of similar image-text pairs, representations of CC3M and CC12M are slightly overlapped. It demonstrates that the model is driven to separate representations from different datasets, and, within a training batch, the model will easily distinguish negative samples.

**Dataset bias influences the optimization of InfoNCE.** Since the dual-encoder model is optimized by InfoNCE loss, we first formulate the loss function and its gradient for further explorations:

$$\mathcal{L} = \sum_j \sum_k y_{jk} \log \left( \frac{\exp(s_{jk})}{\sum_l \exp(s_{jl})} \right), \nabla_\theta \mathcal{L} = - \sum_j \sum_k y_{jk} \nabla_\theta \log \left( \frac{\exp(s_{jk})}{\sum_l \exp(s_{jl})} \right), \tag{1}$$

where the similarity between the *query* $j$ and the *key* $k$ as $s_{jk}$. The ground-truth label corresponding to $s_{jk}$ is represented by $y_{jk} \in \{0, 1\}$. We omit the temperature parameter for simplification. Then, we derive the gradient item as [3]:

$$\begin{aligned} \nabla_\theta \mathcal{L} &= \sum_j \sum_k \left( \frac{\exp(s_{jk})}{\sum_l \exp(s_{jl})} - y_{jk} \right) \nabla_\theta s_{jk} \\ &= \sum_j \sum_k \left( \bar{p}_{jk} - y_{jk} \right) \nabla_\theta s_{jk} \end{aligned} \tag{2}$$

where we could observe that the gradient term is related to the stop-gradient term $\bar{p}_{jk}$, which reflects the similarities among training samples. Negative pairs are essential for self-supervised learning methods which are based on the InfoNCE loss [32]. However, as suggested in Figure 3, dataset bias makes the model easily separate negative samples from different data sources, resulting in the small $\bar{p}_{jk}$ and inferior gradient for negative pairs. Thus, the effectiveness of negative samples are damaged in the optimization process, especially for significant hard examples.

**Debiased sampling.** Knowledge of the dataset bias is not beneficial for downstream tasks and can be even harmful for learning essential semantic concepts. To tackle the dataset bias issue, the key factor is forcing the model to focus on helpful knowledge instead of the dataset bias. Inspired by this, we propose the debiased sampling strategy, as illustrated in Figure 2. Debiased sampling ensures

[3] Detailed deriviations are attached in Appendix A.1.

instances within each batch come from the same dataset. For example, the first batch consists of samples from only SBU, and the second batch is composed of samples of only CC3M. Under this regularization, models are not allowed to hack the optimization by remembering the dataset bias. As shown in Figure 3, the biased distributions of representations are significantly alleviated by our method, especially on the text modality. Besides, as shown in Figure 4, it could be observed that training with debiased sampling yields larger $\bar{p}_{jk}$ of negative pairs (on all datasets) than random sampling, *i.e.*, debiased sampling successfully increases the effectiveness of negative samples. Figure 4 suggests that samples in smaller datasets could suffer from less effective gradient of negative samples, and our method alleviates this problem by increasing gradient of negative samples, especially for small datasets (*i.e.*, VG and SBU). Moreover, downstream results are remarkably improved by the debiased sampling, which will be discussed later.

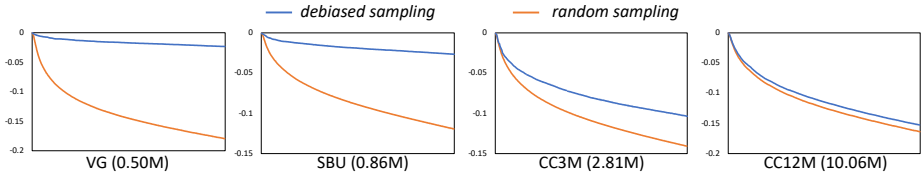


**Fig. 4.** $\log(\bar{p}_{jk})$ averaged over negative pairs on different datasets and scales. The larger value contributes to the larger gradient of negative samples.

### 3.2 Creating Extra Data with Coin Flipping Mixup

Intuitively, data augmentation is a ubiquitous method to create extra training data. With limited data resources, the augmentation plays an important rule in boosting performances. This part introduces a novel data augmentation method, which bring in little computational complexity but remarkably improve model performance.

**Coin flipping mixup.** To the best of our knowledge, mixup [49,42,47,15,26] are seldom investigated in the vision-language pre-training task. In this part, we first formulate the common mixup strategy in the dual-encoder training scheme, and reveal the label assignment dilemma when calculating contrastive loss. To solve this dilemma, we further propose a novel coin flipping mixup.

*(1) Formulations and the label assignment dilemma.* We follow the previous works [49,15] by applying instance-level mixup. Given a batch of N image-text pairs, the image and text of the $j$-th pair are denoted by $I_j$ and $T_j$, respectively. Instead of randomly mixing image-text pairs within the batch, we leverage a more efficient mixing operation for easy implementations:

$$\begin{aligned} \tilde{I}_j &= \lambda * I_j + (1-\lambda) * I_{N-j}, \\ \tilde{T}_j &= \lambda * T_j + (1-\lambda) * T_{N-j}, \end{aligned} \tag{3}$$

where $\tilde{I}_j$ and $\tilde{T}_j$ denote the $j$-th mixed image and text. $\lambda$ is sampled from the distribution $Beta(\alpha, \alpha)$. Therefore, the training batch after the mixing operation could be denoted by $\{(\tilde{I}_1, \tilde{T}_1), (\tilde{I}_2, \tilde{T}_2), \ldots, (\tilde{I}_N, \tilde{T}_N)\}$. However, we will encounter a label assignment dilemma. For instance, both $(\tilde{I}_j, \tilde{T}_j)$ and $(\tilde{I}_{N-j}, \tilde{T}_{N-j})$ are contained in the batch but interpolated by the same instances. It is not feasible to measure the target matching score between $\tilde{I}_j$ and $\tilde{T}_{N-j}$. Particularly, the $\tilde{I}_j$ and $\tilde{T}_{N-j}$ are written as:

$$\begin{aligned} \tilde{I}_j &= \lambda * I_j + (1-\lambda) * I_{N-j}, \\ \tilde{T}_{N-j} &= (1-\lambda) * T_j + \lambda * T_{N-j}, \end{aligned} \tag{4}$$

where the similarity between $\lambda * I_j$ and $(1-\lambda) * T_j$ is not measurable based on the prior knowledge of mixup [49].

*(2) Coin flipping mixup.* To tackle the above problem, we propose the coin flipping mixup strategy. Briefly, mixup is applied on *one of the multiple modals* in each training batch, avoiding the above label assignment dilemma. In our implementation, by uniformly sampling $\gamma$ from the range $[0, 1]$, we enable the mixup on image modal if $\gamma > 0.5$, otherwise text modal. Interestingly, as shown in Figure 5, the strategy is similar to the coin flipping decision-making procedure, from which its name derives.

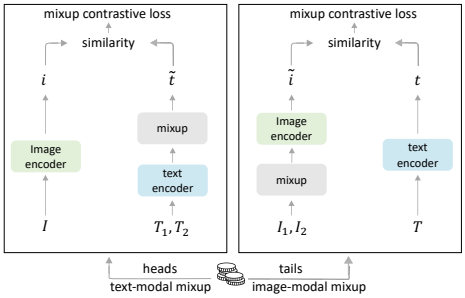


**Fig. 5.** Illustration of our proposed coin flipping mixup. Note that manifold mixup is applied on the text modality, since we empirically observe that interpolating sparse word embeddings could lead to significant performance drop.

We briefly formulate the learning objective of coin flipping mixup, assuming $\gamma > 0.5$ and the mixup on image modal is enabled. In literature [35,22], the contrastive loss could be disentangled to image-to-text and text-to-image matching parts. Correspondingly, the mixup contrastive loss of image-to-text matching is written as:

$$\begin{aligned} \mathcal{L}_{\tilde{I}2T} = \lambda * & \left( -\frac{1}{N} \sum_{j=1}^{N} \log \frac{\exp(\tilde{i}_j \cdot t_j)}{\sum_{k=1}^{N} \exp(\tilde{i}_j \cdot t_k)} \right) \\ + (1-\lambda) * & \left( -\frac{1}{N} \sum_{j=1}^{N} \log \frac{\exp(\tilde{i}_j \cdot t_{N-j})}{\sum_{k=0}^{N-1} \exp(\tilde{i}_j \cdot t_{N-k})} \right), \end{aligned} \tag{5}$$

where $\tilde{i}_j$ and $t_j$ respectively denote representations of the mixed image $\tilde{I}_j$ and the non-mixed text $T_j$. The text-to-image matching part shares similar formulations.

### 3.3 Experiment Results and Discussions

Main results of debiased sampling and coin flipping mixup are reported in Table 3. Overall speaking, both methods benefit performances on both F30K and MS-COCO. Note that these experiments only involve 14M academic data. Stacking

these methods jointly contributes to +31.2 and +35.9 RSUM improvements on F30K and MS-COCO, respectively. Undoubtedly, properly leveraging limited data is of vital importance, and our methods are beneficial.

| | MS-COCO (5K test set) | | | | | Flickr30K (1K test set) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | I → T | | T → I | | | I → T | | T → I | | |
| | R@1 | R@10 | R@1 | R@10 | RSUM | R@1 | R@10 | R@1 | R@10 | RSUM |
| baseline | 45.9 | 82.8 | 35.0 | 73.1 | 371.6 | 66.0 | 95.1 | 58.6 | 90.6 | 483.3 |
| + debiased sampling | 53.2 | 86.4 | 36.7 | 74.1 | 392.3 | 78.8 | 98.2 | 61.2 | 91.9 | 510.1 |
| + coin flipping mixup | 53.0 | 87.6 | 39.6 | 76.5 | 402.8 | 80.1 | 98.4 | 63.7 | 93.1 | 519.2 |

**Table 3.** Results of stacking methods for training with limited data resource.

**Effect of debiased sampling.** Compared to the baseline, debiased sampling achieves consistent and remarkable improvements on all metrics, without any extra computational costs and hyper-parameters. It validates the effectiveness of our proposed debiased sampling, and debiased learning is a potential research direction in the contrastive language-image pre-training field.

**Effect of coin flipping mixup.** We set the alpha value of the beta distribution to 0.1, then apply input mixup on image modal and manifold mixup on text modal. Noticeable promotions are contributed by the coin flipping mixup, especially on text-to-image (T2I) metrics, *i.e.*, text-to-image Recall@1 on F30K and MS-COCO are improved by +2.5 and +2.9.

**Empirical observations on data augmentation.** (1) The cropping area of RandomResizeCrop should be in a relatively large range for covering main objects. (2) AutoAugment [8] brings in satisfactory improvements but little computational overhead. (3) Randomly masking input words advances the model performance with no cost.

## 4 Training with Limited Computational Resource

In contrastive self-supervised learning [5,6], distributed large-batch training has become a standard practice, for increasing the training batch size and providing enough negative samples. Firstly, we demonstrate the remarkable benefits of distributed large-batch training in the multi-modal pre-training task; however, it relies on considerable computational resources (*e.g.*, training our model with 16,384 batch size needs 128 V100 GPUs). Then, to tackle this problem, we study how to achieve comparable results with limited computational resources (*e.g.*, 8 V100 GPUs) by proposing the decoupled gradient accumulation. Lastly, we discuss how to accelerate the training.

### 4.1 Large-Batch Training with Decoupled Gradient Accumulation

**Benefits of large-batch distributed training.** In the practical implementation of distributed InfoNCE loss, gather operations are frequently used to collect negative samples across machines. In multi-modal scenario, the InfoNCE loss

| | MS-COCO (5K test set) | | | | | Flickr30K (1K test set) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | I → T | | T → I | | | I → T | | T → I | | |
| | R@1 | R@10 | R@1 | R@10 | RSUM | R@1 | R@10 | R@1 | R@10 | RSUM |
| baseline + data | 53.0 | 87.6 | 39.6 | 76.5 | 402.8 | 80.1 | 98.4 | 63.7 | 93.1 | 519.2 |
| + gradient reserved gather | 55.4 | 88.7 | 42.0 | 78.7 | 415.0 | 81.4 | 98.2 | 66.2 | 93.7 | 524.1 |
| + batch = 2,048 | 56.4 | 88.5 | 42.7 | 79.2 | 418.0 | 81.5 | 98.6 | 68.2 | 93.7 | 527.5 |
| + batch = 4,096 | 58.9 | 89.9 | 43.8 | 79.6 | 425.9 | 82.7 | 98.6 | 68.7 | 94.5 | 531.7 |
| + batch = 8,192 | 59.0 | 89.5 | 43.7 | 79.5 | 424.4 | 83.1 | 98.7 | 68.5 | 94.6 | 531.8 |
| + batch = 16,384 | 59.3 | 89.6 | 44.1 | 70.4 | 425.0 | 85.5 | 98.5 | 69.8 | 94.5 | 536.2 |

**Table 4.** Results of distributed large-batch training. "baseline + data" denotes the result of stacking methods proposed in Sec. 3.

could be separated into image-to-text (I2T) and text-to-image (T2I) matching parts. Similar to Eqn. (2), the gradient of the I2T part is as followed [4]:

$$\nabla_\theta \mathcal{L}^{I2T} = \sum_j \sum_k \left(\bar{p}_{jk}^{I2T} - y_{jk}^{I2T}\right) \left(\bar{i}_j \nabla_\theta t_k + \bar{t}_k \nabla_\theta i_j\right), \tag{6}$$

where we place a vinculum on a value to indicate its gradient is detached. For a pair $(i_j, t_k)$ from *different* machines, gather operations with detaching gradients would produce the following wrong gradient on the machine of sample $j$:

$$\tilde{\nabla}_\theta \mathcal{L}_{ij}^{I2T} = \left(\bar{p}_{jk}^{I2T} - y_{jk}^{I2T}\right) \bar{t}_k \nabla_\theta i_j. \tag{7}$$

Therefore, preserving gradients of gathered embeddings would provide valuable gradients. As reported in Table 4, by preserving gradients of gathered embeddings, noticeable gains are achieved within expectations. Concretely, +4.9 RSUM on F30K and +12.2 RSUM on MS-COCO are contributed by the gradient reversed gather, further supporting our derivations.

Previous works have demonstrated that self-supervised contrastive learning could significantly benefit from the large training batch size, which provides more negative examples to facilitate the model convergence [5]. To further analyze the impact of varying batch sizes on multi-modal contrastive pre-training, we scale the batch size from 1,024 to 16,384 and keep training epochs consistent. Besides, previous works [5,17] empirically showed that linearly scaling the initial learning rate is necessary for large-batch training. Regarding large batch experiments, up to 128 Nvidia V100 GPUs are used. As shown in Table 4, increasing the batch size from 1,024 to 16,384 leads to significant improvements on all evaluated metrics, indicating the vital importance of large-batch training. However, substantial computational resources are used for containing large batches.

**Decoupled gradient accumulation.** A common strategy to mimic large-batch training is the multi-step gradient accumulation. Concretely, a training iteration of a large batch is divided into several sub-iterations, and, in each sub-iteration, the batch size is relatively small. Gradients of multiple sub-iterations are individually calculated, accumulated and jointly back-propagated. It is a practical strategy in deep learning tasks; however, to mimic the large batch InfoNCE loss, the calculation process unavoidably involves embeddings from

[4] Due to the page limit, detailed formulations are attached in Appendix A.1.

different training sub-iterations, which are, unfortunately, not accessible across sub-iterations. Therefore, the conventional multi-step gradient accumulation is not able to enlarge the effective batch size, greatly limiting final model performances.

We propose the decoupled gradient accumulation to make large-batch contrastive learning feasible for limited resources. According to Eqn. (6), we mathematically decouple the gradient of a large batch into two parts [5]:

$$\begin{aligned}\nabla_\theta \mathcal{L} &= \nabla_\theta \mathcal{L}^{I2T} + \nabla_\theta \mathcal{L}^{T2I} \\ &= \sum_j \nabla_\theta \underbrace{\left( \sum_k \left( \bar{p}_{jk}^{I2T} - y_{jk}^{I2T} + \bar{p}_{kj}^{T2I} - y_{kj}^{T2I} \right) \bar{t}_k \right)}_{\text{stop-gradient part}} i_j \\ &\quad + \sum_k \nabla_\theta \underbrace{\left( \sum_j \left( \bar{p}_{jk}^{I2T} - y_{jk}^{I2T} + \bar{p}_{kj}^{T2I} - y_{kj}^{T2I} \right) \bar{i}_j \right)}_{\text{stop-gradient part}} t_k,\end{aligned} \tag{8}$$

where one part of gradient is only related to embeddings within each sub-iteration, and the other part only depends on stop-gradient embeddings of the large batch, which can be obtained by forwarding the large batch for an extra time. In this manner, we are allowed to take advantages of the large batch size by sacrificing training time.

As reported in Table 5, it empirically shows that, by sacrificing extra 40%–50% training time, our gradient accumulation successfully mimics large-batch training without damaging model performances. With 8 V100 GPUs, we are not allowed to train the model with batch sizes larger than 1,024, and thus achieved performances are relatively unsatisfactory. However, our method successfully allows us to train models with large effective batch sizes 8,192 and 16,384, achieving comparable RSUM scores with only 8 V100 GPUs.

| batch | DGA step | effective batch | # GPU | GPU time (hr) | MS-COCO RSUM | F30K RSUM |
|---|---|---|---|---|---|---|
| 1,024 | – | 1,024 | 8 | ∼430 | 415.0 | 524.1 |
| 8,192 | – | 8,192 | 64 | – | 424.4 | 531.8 |
| 16,384 | – | 16,384 | 128 | – | 425.0 | 536.2 |
| 1,024 | 8 | 8,192 | 8 | ∼600 | 424.1 | 532.2 |
| 1,024 | 16 | 16,384 | 8 | ∼680 | 425.2 | 535.9 |

**Table 5.** RSUM scores of decoupled gradient accumulation (DGA). For training with batch 8,192 and 16,384, 64 and 128 V100 GPUs are required, respectively.

**Stable decoupled gradient accumulation.** Note that encoders could contain modules of randomness, *e.g.*, dropout layers are widely applied in the BERT [10]. Thus, forwarding the same sample two times could produce different embeddings. To this end, we set the identical random seed for twice forwarding processes, eliminating the randomness and stabilizing the training. In Table 6, we provide an ablation study related to the stable training. It demonstrates that significant

[5] Due to the page limit, detailed derivations are attached in Appendix A.2.

performance drops would be caused without considering the randomness. Forwarding the same sample for two times yields different embeddings results in the gradient in Eqn.(8) is wrongly calculated.

| batch | DGA step | effective batch | # GPU | **stable** | MS-COCO RSUM | F30K RSUM |
|---|---|---|---|---|---|---|
| 1,024 | 16 | 16,384 | 8 | ✓ | 425.2 | 535.9 |
| 1,024 | 16 | 16,384 | 8 | – | 413.4 | 527.1 |

**Table 6.** Effects of stable training. "✓" denotes setting the identical random seed for twice forwarding processes, and the achieved results correspond to Table 5.

### 4.2 Fast Training with TokenDrop and Auxiliary Encoders

Thus far, all methods for better performances are elaborated. For real-scenario multi-modal applications, the training efficiency and model performance are equally significant for various deployment purposes. We introduce two methods on fast training for different purposes.

**TokenDrop.** Inspired by the recent work [16], we randomly drop a part of input pixels to speed-up the training of image encoders. Empirically, we observe that randomly masking 25% input tokens of ViT introduces negligible performance drop, but considerably reduces training time. As shown in Table 7, enabling TokenDrop saves ∼30% training time. Besides, training with TokenDrop compensates for the extra training time caused by DGA.

| batch | DGA step | Token Drop | # GPU | GPU time (hr) | MS-COCO RSUM | F30K RSUM |
|---|---|---|---|---|---|---|
| 1,024 | 16 | – | 8 | ∼680 | 425.2 | 535.9 |
| 1,024 | 16 | ✓ | 8 | ∼**470** | 424.8 | 535.5 |

**Table 7.** Training time saved by TokenDrop.

**Auxiliary Encoders.** Assuming that we have trained a model with heavy encoders, we investigate how to fast obtain lightweight encoders with auxiliary heavy ones. Since the training of a dual-encoder model is driven by the InfoNCE loss, embeddings yielded by either encoder are regarded as the "learning target" of the other side. Thus, enlarging either encoder's capacity would contribute to more reliable and discriminative embeddings. Assuming that we have trained a dual-encoder model with heavy encoders, *e.g.*, ViT-B/16 and BERT-Base, we can replace one of them to a lightweight one and re-train it with the guidance of the other one in a distillation manner [19,52]. For instance, we change the image encoder from ViT-B/16 to ViT-B/32, and then re-train it with the BERT-Base being frozen. With the guidance of the frozen encoder, the training process of the replaced encoder could be greatly accelerated, as reported in Table 8.

| training method | encoder image | encoder text | GPU time (hr) | MS-COCO RSUM | F30K RSUM |
|---|---|---|---|---|---|
| auxiliary | ViT-B/16 | BERT-B | – | 402.8 | 519.1 |
| | ViT-B/32 | BERT-B♠ | ∼**110** | 381.2 | 493.9 |
| baseline | ViT-B/32 | BERT-B | ∼240 | 379.5 | 494.1 |

**Table 8.** Training time saved by the auxiliary encoder method. "♠" symbol denotes the model is frozen.

## 5 Comparisons with SOTA Methods

In this section, we focus on assessing the pre-training performances with cross-modal retrieval tasks, in both zero-shot and fine-tuned settings [35,22]. We name our method as "ZeroVL", where "Zero" means the motivation for designing a strong baseline with limited resources.

### 5.1 Zero-Shot Cross-Modal Retrieval

| | computation device | count | data | input size | batch size | MS-COCO (5K test set) I → T R@1 | I → T R@10 | T → I R@1 | T → I R@10 | RSUM | Flickr30K (1K test set) I → T R@1 | I → T R@10 | T → I R@1 | T → I R@10 | RSUM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***zero-shot*** | | | | | | R@1 | R@10 | R@1 | R@10 | RSUM | R@1 | R@10 | R@1 | R@10 | RSUM |
| CLIP | V100 | 256 | 400M | 336 | 32,768 | 58.4 | 88.1 | 37.8 | 72.2 | 400.2 | 88.0 | 99.4 | 68.7 | 95.2 | 540.6 |
| ALIGN | $TPU_{v3}$ | 1,024 | 1800M | 289 | 16,384 | 58.6 | 89.7 | 45.6 | 78.6 | 425.3 | **88.6** | **99.7** | **75.7** | **96.8** | **553.3** |
| baseline | V100 | 8 | 14M | 224 | 1,024 | 45.9 | 82.8 | 35.0 | 73.1 | 371.6 | 66.0 | 95.1 | 58.6 | 90.6 | 483.3 |
| CLIP (our impl.) | V100 | 8 | 14M | 224 | 1,024 | 51.0 | 85.5 | 38.2 | 75.5 | 392.5 | 80.9 | 97.8 | 63.8 | 92.4 | 518.4 |
| CLIP (our impl.) | V100 | 128 | 14M | 224 | 16,384 | 57.7 | 88.7 | 41.6 | 77.8 | 416.0 | 83.1 | 98.3 | 67.2 | 93.9 | 527.3 |
| ZeroVL (ours) | V100 | 8 | 14M | 224 | 16,384 | 59.3 | 89.6 | 44.1 | 79.5 | 425.0 | 85.5 | 98.5 | 69.8 | 94.5 | 536.2 |
| ZeroVL† (ours) | V100 | 8 | 100M | 224 | 16,384 | **64.0** | **91.4** | **47.3** | **81.1** | **442.1** | 88.0 | 99.2 | 73.5 | 95.7 | 546.5 |

**Table 9.** *Zero-shot* cross-modal retrieval results. "baseline" is the naive baseline in Sec. 2. "†" means training with the 100M web data.

**Setup.** Training implementation details are as followed. On the ground of baseline settings (*e.g.*, learning rate, training epoch, and weight decay) introduced in Sec. 2.2, we stack all proposed methods, *i.e.*, debaised sampling, coin flipping mixup, and decoupled gradient accumulation. For reproducibility, we mainly benchmark with publicly accessible academic datasets. For fair comparisons, we re-implement CLIP with 14M data to validate the performance drop caused by limited resources. Besides, CLIP and ALIGN respectively collect 400M and 1.8B image-text pairs from the web. Due to training datasets of CLIP and ALIGN are not available, we also collect ∼100M web image-text pairs for validating the effectiveness of our method on large-scale data.

**Main results.** In Table 9, on both F30K and MS-COCO datasets, we achieve competitive results on the basis of 14M academic publicly accessible data and 8 V100 GPUs. It is worth mentioning that our ZeroVL already exceeds CLIP on the MS-COCO dataset in both image-to-text (I2T) and text-to-image (T2I) metrics, *e.g.*, our I2T R@1 and T21 R@1 surpass CLIP by +0.9 and +6.3, respectively. Results of our implemented CLIP further validate the contribution of our efforts, *i.e.*, the performance of cross-modal retrieval would be greatly suppressed if the

resources were greatly limited. In addition, although our collected 100M web images are much less than those of CLIP and ALIGN, ZeroVL still successfully outperforms CLIP trained with 400M data and ALIGN trained with 1.8B data on MS-COCO. On F30K, we perform slightly worse than ALIGN but better than CLIP, which can result from the domain of ALIGN's data is larger than ours.

**Resource costs.** For computational resources, training CLIP requires 256 V100 GPUs, and training ALIGN needs 1,024 Could TPUv3 cores. Experiments in Table 9 involve 8 V100 32GB GPUs. For data resources, we mainly benchmark on 14M publicly accessible academic datasets to guarantee the reproducibility. Experiments of 100M web data demonstrate that our method is still effective on large-scale data, *i.e.*, our method fits in different data scales without tuning hyper-parameters. Additionally, only 2.4 days are required for training ZeroVL with 8 V100 and 14M academic data, which could be friendly to the most researchers.

### 5.2 Fine-Tuned Cross-Modal Retrieval

| | input | encoder | | MS-COCO (5K test set) | | | | | Flickr30K (1K test set) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | size | image (I) | text (T) | I → T | | T → I | | | I → T | | T → I | | |
| ***fine-tuned*** | | | | R@1 | R@10 | R@1 | R@10 | RSUM | R@1 | R@10 | R@1 | R@10 | RSUM |
| VSE++ | 512 | RX101* | BERT-B | 57.9 | 92.8 | 44.9 | 84.0 | 439.2 | 80.9 | 98.9 | 65.2 | 93.7 | 524.8 |
| GPO | 512 | RX101* | BERT-B | 68.1 | 95.2 | 52.7 | 88.3 | 474.8 | 88.7 | 99.8 | 76.1 | 97.1 | 555.1 |
| ALIGN | 289 | EffNet-L2* | BERT-L | 77.0 | 96.9 | **59.9** | 89.8 | 500.4 | **95.3** | **100.0** | **84.9** | **98.6** | **576.0** |
| baseline | 224 | ViT-B/16 | BERT-B | 69.1 | 94.8 | 51.9 | 86.8 | 471.9 | 90.1 | 99.1 | 75.1 | 96.6 | 553.0 |
| CLIP (our impl. 8V100) | 224 | ViT-B/16 | BERT-B | 69.9 | 94.9 | 52.5 | 87.0 | 473.8 | 90.4 | 99.2 | 75.6 | 96.5 | 554.1 |
| CLIP (our impl. 128V100) | 224 | ViT-B/16 | BERT-B | 71.7 | 95.8 | 54.0 | 88.1 | 481.3 | 91.1 | 99.5 | 78.5 | 97.7 | 560.7 |
| ZeroVL (ours) | 224 | ViT-B/16 | BERT-B | 72.9 | 95.9 | 55.1 | 88.6 | 485.0 | 91.7 | 99.5 | 79.2 | 97.1 | 561.6 |
| ZeroVL† (ours) | 288 | ViT-B/16 | BERT-B | **77.2** | **97.1** | 59.3 | **90.2** | **500.5** | 95.0 | 100.0 | 83.7 | 98.6 | 573.6 |

**Table 10.** *Fine-tuned* cross-modal retrieval results of representative dual-encoder methods. "RX101*" correspond to the ResNeXt-101 model pre-trained on Instagram-1B [45]. "EffNet-L2*" denotes the large CNN model EfficientNet-L2 [41,44]. "†" denotes pre-training with the 100M web data.

**Setup.** After the pre-training phase, we fine-tune the model on downstream datasets F30K and MS-COCO. Fine-tuning hyper-parameters are identical to pre-training's, except the initial learning rate, training epoch, and batch size. The is learning rate is set to 1e-5. For F30K and MS-COCO, we optimize the model for 1K and 5K steps. Batch size is set to 2,048. Similar to zero-shot experiments, we also provide fine-tuning results with both 14M and 100M data.

**Main results.** In Table 10, with 14M academic pre-training data, we successfully outperforms state-of-the-art in-domain training method VSE++ [12] and GPO [4]. It is worth mentioning that GPO also involves large-scale pre-training on the image modal, *i.e.*, weakly supervised pre-training with the Instagram-1B dataset [45]. Compared with GPO, ZeroVL can achieve better results with the more efficient image encoder and smaller training input size, strongly supporting the effectiveness of our pre-training method. For experiments with 100M web data, it is worth noting that ALIGN uses (1) significantly more pre-training data, (2) heavier image and text encoders, and (3) larger pre-training resolutions

than our method. Nevertheless, similar to results in zero-shot, we still achieve comparable results to ALIGN.

### 5.3 Linear Probing

| | pre-training | | | | | linear probing | | |
|---|---|---|---|---|---|---|---|---|
| | computation device | computation count | data | input size | batch size | backbone (#params) | input size | top-1 accuracy |
| CLIP | V100 | 256 | 400M | 224 | 32,768 | ViT-B/16 (87M) | 224 | 80.2 |
| ALIGN | $TPU_{v3}$ | 1,024 | 1800M | 289 | 16,384 | EffNet-L2 (480M) | 600 | 85.5 |
| CLIP (our impl.) | V100 | 8 | 14M | 224 | 1,024 | ViT-B/16 (87M) | 224 | 75.9 |
| CLIP (our impl.) | V100 | 128 | 14M | 224 | 16,384 | ViT-B/16 (87M) | 224 | 80.0 |
| ZeroVL (ours) | V100 | 8 | 14M | 224 | 16,384 | ViT-B/16 (87M) | 224 | **80.9** |

**Table 11.** Linear probing results on ImageNet-1K.

**Setup.** Following [35,22], we conduct the linear probing task on ImageNet-1K [9] after the pre-training phase. The batch size is set to 16,384 and learning rate is set to 6.4. We optimize the model for 90 epochs with the LARS optimizer [46], and weight decay is set to 0. To reveal the effects of our proposed methods on linear probing, we also evaluate the re-implemented CLIP as mentioned above.
**Main results.** In Table 11, ZeroVL out-performs CLIP by 0.7%. However, similar to fine-tuned cross-modal retrieval, ALIGN achieves better results than ZeroVL based on heavier pre-training costs, larger model capacity, and larger image resolutions. Moreover, there are two observations on re-implemented CLIP. Firstly, we observe that training with limited computation resource (8 V100) achieves unsatisfactory top-1 accuracy 75.9%. Secondly, training CLIP with rich computation resource (128 V100) greatly improves the accuracy to 80.0%. The differences between ZeroVL and re-implemented CLIP (with 128 V100) are methods proposed in Sec. 3, validating the effectiveness of our proposed debiased sampling and coin flipping mixup. Benefits of our methods for cutting down the heavy resources dependency are further confirmed.

## 6 Conclusion

This work provides a training guideline for conducting dual-encoder multi-modal representation contrastive learning with limited resources. The proposed methods significantly lower computational resources, while still achieving good performance to be applied in other vision-language downstream tasks. With only 14M publicly accessible academic datasets and 8 V100 GPUs, we provide a reproducible strong baseline. In addition, we achieve comparable or superior performances than state-of-the-art methods with 100M web data. We hope our training pipeline and benchmark will be useful for future researches in the multi-modal representation learning field and benefit the community.

## References


1. Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L., Parikh, D.: VQA: Visual Question Answering. In: ICCV (2015)
2. Changpinyo, S., Sharma, P., Ding, N., Soricut, R.: Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In: CVPR (2021)
3. Chen, H., Ding, G., Liu, X., Lin, Z., Liu, J., Han, J.: Imram: Iterative matching with recurrent attention memory for cross-modal image-text retrieval. In: CVPR (2020)
4. Chen, J., Hu, H., Wu, H., Jiang, Y., Wang, C.: Learning the best pooling strategy for visual semantic embedding. In: CVPR (2021)
5. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML (2020)
6. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.: Big self-supervised models are strong semi-supervised learners. In: NeurIPS (2020)
7. Chen, Y.C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., Liu, J.: UNITER: Universal image-text representation learning. In: ECCV (2020)
8. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: AutoAugment: Learning augmentation strategies from data. In: CVPR (2019)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805 (2018)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
12. Faghri, F., Fleet, D.J., Kiros, J.R., Fidler, S.: VSE++: Improving visual-semantic embeddings with hard negatives. In: BMVC (2018)
13. Gan, Z., Chen, Y.C., Li, L., Zhu, C., Cheng, Y., Liu, J.: Large-scale adversarial training for vision-and-language representation learning. In: NeurIPS (2020)
14. Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., Parikh, D.: Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In: CVPR (2017)
15. Guo, H., Mao, Y., Zhang, R.: Augmenting data with mixup for sentence classification: An empirical study. arXiv:1905.08941 (2019)
16. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. arXiv:2111.06377 (2021)
17. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR (2020)
18. He, P., Liu, X., Gao, J., Chen, W.: Deberta: Decoding-enhanced bert with disentangled attention. In: ICLR (2021)
19. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv:1503.02531 (2015)
20. Huang, Z., Zeng, Z., Huang, Y., Liu, B., Fu, D., Fu, J.: Seeing Out of tHe bOx: End-to-end pre-training for vision-language representation learning. In: CVPR (2021)
21. Jegou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. TPAMI (2010)

22. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q.V., Sung, Y., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: ICML (2021)
23. Kim, W., Son, B., Kim, I.: ViLT: Vision-and-language transformer without convolution or region supervision. In: ICML (2021)
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
25. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual Genome: Connecting language and vision using crowdsourced dense image annotations. In: IJCV (2017)
26. Lee, K., Zhu, Y., Sohn, K., Li, C.L., Shin, J., Lee, H.: i-mix: A domain-agnostic strategy for contrastive representation learning. In: ICLR (2021)
27. Li, J., Selvaraju, R.R., Gotmare, A.D., Joty, S., Xiong, C., Hoi, S.: Align before fuse: Vision and language representation learning with momentum distillation. In: NeurIPS (2021)
28. Li, L.H., Yatskar, M., Yin, D., Hsieh, C.J., Chang, K.W.: VisualBERT: A simple and performant baseline for vision and language. arXiv:1908.03557 (2019)
29. Li, X., Yin, X., Li, C., Zhang, P., Hu, X., Zhang, L., Wang, L., Hu, H., Dong, L., Wei, F., et al.: OSACR: Object-semantics aligned pre-training for vision-language tasks. In: ECCV (2020)
30. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv:1907.11692 (2019)
31. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
32. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv:1807.03748 (2018)
33. Ordonez, V., Kulkarni, G., Berg, T.: Im2text: Describing images using 1 million captioned photographs. In: NeurIPS (2011)
34. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: StyleCLIP: Text-driven manipulation of stylegan imagery. In: ICCV (2021)
35. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML (2021)
36. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual Captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: ACL (2018)
37. Su, W., Zhu, X., Cao, Y., Li, B., Lu, L., Wei, F., Dai, J.: VL-BERT: Pre-training of generic visual-linguistic representations. arXiv:1908.08530 (2019)
38. Suhr, A., Lewis, M., Yeh, J., Artzi, Y.: A corpus of natural language for visual reasoning. In: ACL (2017)
39. Suhr, A., Zhou, S., Zhang, A., Zhang, I., Bai, H., Artzi, Y.: A corpus for reasoning about natural language grounded in photographs. In: ACL (2019)
40. Sun, C., Myers, A., Vondrick, C., Murphy, K., Schmid, C.: VideoBERT: A joint model for video and language representation learning. In: ICCV (2019)
41. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML (2019)
42. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: ICML (2019)
43. Wu, H., Mao, J., Zhang, Y., Jiang, Y., Li, L., Sun, W., Ma, W.Y.: Unified visual-semantic embeddings: Bridging vision and language with structured meaning representations. In: CVPR (2019)

44. Xie, Q., Luong, M.T., Hovy, E., Le, Q.V.: Self-training with noisy student improves imagenet classification. In: CVPR (2020)
45. Yalniz, I.Z., Jégou, H., Chen, K., Paluri, M., Mahajan, D.: Billion-scale semi-supervised learning for image classification. arXiv:1905.00546 (2019)
46. You, Y., Gitman, I., Ginsburg, B.: Large batch training of convolutional networks. arXiv:1708.03888 (2017)
47. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: CutMix: Regularization strategy to train strong classifiers with localizable features. In: ICCV (2019)
48. Zellers, R., Bisk, Y., Farhadi, A., Choi, Y.: From recognition to cognition: Visual commonsense reasoning. In: CVPR (2019)
49. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: Mixup: Beyond empirical risk minimization. In: ICLR (2018)
50. Zhang, P., Goyal, Y., Summers-Stay, D., Batra, D., Parikh, D.: Yin and Yang: Balancing and answering binary visual questions. In: CVPR (2016)
51. Zhang, P., Li, X., Hu, X., Yang, J., Zhang, L., Wang, L., Choi, Y., Gao, J.: VinVL: Revisiting visual representations in vision-language models. In: CVPR (2021)
52. Zhao, B., Cui, Q., Song, R., Qiu, Y., Liang, J.: Decoupled knowledge distillation. In: CVPR (2022)