Sigmoid Loss for Language Image Pre-Training

Xiaohua Zhai*  Basil Mustafa  Alexander Kolesnikov  Lucas Beyer*
Google DeepMind, Zürich, Switzerland
{xzhai, basilm, akolesnikov, lbeyer}@google.com

Abstract

We propose a simple pairwise sigmoid loss for image-text pre-training. Unlike standard contrastive learning with softmax normalization, the sigmoid loss operates solely on image-text pairs and does not require a global view of the pairwise similarities for normalization. The sigmoid loss simultaneously allows further scaling up the batch size, while also performing better at smaller batch sizes. With only four TPUv4 chips, we can train a Base CLIP model at 4 k batch size and a Large LiT model at 20 k batch size, the latter achieves 84.5% ImageNet zero-shot accuracy in two days. This disentanglement of the batch size from the loss further allows us to study the impact of examples vs pairs and negative to positive ratio. Finally, we push the batch size to the extreme, up to one million, and find that the benefits of growing batch size quickly diminish, with a more reasonable batch size of 32 k being sufficient. We hope our research motivates further explorations in improving the quality and efficiency of language-image pre-training.

1. Introduction

Contrastive pre-training using weak supervision from image-text pairs found on the web is becoming the go-to method for obtaining generic computer vision backbones, slowly replacing pre-training on large labelled multi-class datasets. The high-level idea is to simultaneously learn an aligned representation space for images and texts using paired data. Seminal works CLIP [34] and ALIGN [23] established the viability of this approach at a large scale, and following their success, many large image-text datasets became available privately [55, 13, 21, 45] and publicly [37, 6, 15, 7, 38].

The standard recipe to pre-train such models leverages the image-text contrastive objective. It aligns the image and text embeddings for matching (positive) image-text pairs while making sure that unrelated (negative) image-text pairs are dissimilar in the embedding space. This is achieved via a batch-level softmax-based contrastive loss, applied twice to normalize the pairwise similarity scores across all images, then all texts. A naive implementation of the softmax is numerically unstable; it is usually stabilized by subtracting the maximum input value before applying the softmax [18], which requires another pass over the full batch.

In this paper, we propose a simpler alternative: the sigmoid loss. It does not require any operation across the full batch and hence greatly simplifies the distributed loss implementation and boosts efficiency. Additionally, it conceptually decouples the batch size from the definition of the task. We compare the proposed sigmoid loss with the standard softmax loss across multiple setups. In particular, we investigate sigmoid-based loss with two prominent approaches for image-text learning: CLIP [34] and LiT [55], which we call sigmoid language image pre-

<table>
<thead>
<tr>
<th>Image</th>
<th>Text</th>
<th>BS</th>
<th>#TPUv4</th>
<th>Days</th>
<th>INet-0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SigLiT</td>
<td>B/8</td>
<td>L</td>
<td>32 k</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>SigLiT</td>
<td>g/14</td>
<td>L</td>
<td>20 k</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>SigLiP</td>
<td>B/16</td>
<td>B</td>
<td>16 k</td>
<td>16</td>
<td>3</td>
</tr>
<tr>
<td>SigLiP</td>
<td>B/16</td>
<td>B</td>
<td>32 k</td>
<td>32</td>
<td>2</td>
</tr>
<tr>
<td>SigLiP</td>
<td>B/16</td>
<td>B</td>
<td>32 k</td>
<td>32</td>
<td>5</td>
</tr>
</tbody>
</table>

* We use a variant of the L model with 12 layers.
training (SigLIP) and sigmoid LiT (SigLiT), respectively. We find that the sigmoid loss performs significantly better than the softmax loss when the batch size is smaller than 16 k. As the train batch size grows, the gap closes. Importantly, the sigmoid loss is symmetric, requires just a single pass, and a typical implementation requires less memory than the softmax loss. This enables successful training of a SigLiT model at a batch size of one million. However, we find that the performance saturates with growing batch size, both for softmax and sigmoid. The good news is that a reasonable batch size, i.e. 32 k, is sufficient for image-text pre-training. This conclusion also holds for multilingual SigLIP training on over 100 languages.

In Table 1, we present setups for image-text pre-training that require a moderate amount of TPUv4 chips for training. SigLiT is surprisingly efficient, reaching 79.7% zero-shot accuracy on ImageNet in just a single day on four chips. SigLIP’s more demanding from-scratch training reaches 73.4% zero-shot accuracy in 5 days with 32 TPUv4 chips. This compares favorably to prior works such as FLIP [29] and CLIP [34], which require approximately 5 and 10 days respectively on 256 TPUv3 cores. When fine-tuning a pre-trained vision backbone in SigLIP, denoted as SigLIT, we found that disabling the weight decay on the pre-trained backbone leads to better results (see Figure 4 for details). We hope our work paves the way for making the nascent language-image pre-training field more accessible.

2. Related Work

Contrastive learning with the sigmoid loss. One prior work proposes a similar sigmoid loss for the task of unsupervised dimensionality reduction [19]; in the scope of contrastive image-text learning, the vast majority of works rely on the softmax-based InfoNCE loss as popularized by [43]. In supervised classification, the sigmoid loss has already been shown to be slightly more effective and robust than the softmax loss [3, 47].

Contrastive language-image pre-training has become popular since CLIP [34] and ALIGN [23] applied softmax contrastive learning [56, 43, 10, 24] to large-scale image-text datasets. Both models perform very well on zero-shot transfer tasks, including classification and retrieval. Follow-up works show that contrastively pre-trained models produce good representations for fine-tuning [49, 16], linear regression [23], object detection [30], semantic segmentation [31] and video tasks [53].

Generative language-image pre-training Besides softmax contrastive pre-training, various alternatives have been proposed. GIT [45], SimVLM [46], and LEMON [21] successfully pre-train models using a generative text decoder instead, while CoCa [52] adds such a decoder to the discriminative CLIP/ALIGN setup, thus combining the pros and cons of both approaches into a single very capable model. BLIP [28] further proposes CapFilt which uses the generative decoder to create better captions and the discriminative part of the model to filter pairs. Language-Image pre-training is a very active field and surveys [8] rapidly become outdated.

Efficient language-image pre-training On the other hand, few works have tried making language image-pre-training more efficient. LiT [55] and FLIP [29] are notable attempts, the former requires a pre-trained and locked backbone, and the latter sacrifices quality by randomly dropping visual tokens. BASIC [33] and LAION [48] look at scaling batch-size but only go up to 16 k and 160 k respectively, by using many hundreds of chips, and for the former also mixing in a large private classification dataset [33, 51]. The recent Lion optimizer [12] claims to be able to reduce the training cost to reach similar quality.

3. Method

In this section, we first review the widely-used softmax-based contrastive loss. We then introduce the pairwise sigmoid loss and discuss its efficient implementation.

Given a mini-batch \( B = \left\{ (I_1, T_1), (I_2, T_2), \ldots \right\} \) of image-text pairs, the contrastive learning objective encourages embeddings of matching pairs \((I_i, T_i)\) to align with each other, while pushing embeddings of unmatched pairs \((I_i, T_j \neq i)\) apart. For practical purposes, it is assumed that for all images \( i \), the text associated with a different image \( j \), and vice-versa. This assumption is usually noisy and imperfect.

3.1. Softmax loss for language image pre-training

When using the softmax loss to formalize this objective, an image model \( f(x) \) and a text model \( g(y) \) are trained to minimize the following objective:

\[
- \frac{1}{2|B|} \sum_{i=1}^{|B|} \left( \log \frac{e^{f(x_i) \cdot y_i}}{\sum_{j=1}^{|B|} e^{f(x_i) \cdot y_j}} + \log \frac{e^{f(x_i) \cdot y_i}}{\sum_{j=1}^{|B|} e^{f(x_i) \cdot y_j}} \right)
\]
(a) Initially each device holds 4 image and 4 text representations. Each device needs to see the representations from other devices to calculate the full loss.

(b) They each compute the component of the loss (highlighted) for their representations, which includes the positives. This makes sure the training starts roughly close to the prior and does not require massive over-correction. Algorithm 1 presents a pseudocode implementation of the proposed sigmoid loss for language image pre-training.

(c) Texts are swapped across the devices, so device 1 now has $T_{1.4}$ & text pair have interacted, e.g. device 1 has the loss of $I_{1.4}$ and $T_{1,12}$. A final cross-device sum brings everything together.

(d) This repeats till every image and text pair independently, effectively turning the learning problem into the standard binary classification.

3.2. Sigmoid loss for language image pre-training

Instead of the softmax-based contrastive loss, we propose a simpler alternative that does not require computing global normalization factors. The sigmoid-based loss processes every image-text pair independently, effectively turning the learning problem into the standard binary classification on the dataset of all pair combinations, with a positive label for the matching pairs $(I_i, T_i)$ and negative labels for all other pairs $(I_i, T_j \neq i)$. It is defined as follows:

$$
-\frac{1}{|B|} \sum_{i=1}^{|B|} \sum_{j=1}^{|B|} \log \left( 1 + e^{-(t \cdot x_i + y_j + b)} \right)
$$

where $z_{ij}$ is the label for a given image and text input, which equals 1 if they are paired and -1 otherwise. Note that at initialization, the heavy imbalance coming from the many negatives dominates the loss, leading to large initial optimization steps trying to correct this bias. To alleviate this, we introduce an additional learnable bias term $b$ similar to the temperature $t$. We initialize $t'$ and $b$ to 10 and -10 respectively. This makes sure the training starts roughly close to the prior and does not require massive over-correction. Algorithm 1 presents a pseudocode implementation of the proposed sigmoid loss for language image pre-training.

3.3. Efficient “chunked” implementation

Contrastive training typically utilizes data parallelism. Computing the loss when data is split across $D$ devices necessitates gathering all embeddings [55] with expensive all-gathers and, more importantly, the materialization of a memory-intensive $|B| \times |B|$ matrix of pairwise similarities.

The sigmoid loss, however, is particularly amenable to a memory efficient, fast, and numerically stable implementation that ameliorates both these issues. Denoting the per-device batch size as $b = \frac{|B|}{D}$, the loss is reformulated as:

$$
-1 \frac{1}{|B|} \sum_{d_i=1}^D \sum_{b_{d_i+1}}^{b_{d_i+1}} \sum_{i=b_{d_i}}^{b_{d_i+1}} \sum_{j=b_{d_j}}^{b_{d_j+1}} \mathcal{L}_{ij}
$$

This is particularly simple for the sigmoid loss as each pair is an independent term in the loss. Figure 1 illustrates this method. In words, we first compute the component of the loss corresponding to the positive pairs, and $b - 1$ negative pairs. We then permute representations across devices, so each device takes negatives from its neighbouring device (next iteration of sum $B$). The loss is then calculated with respect to this chunk (sum $C$). This is done independently in each device, such that each device computes the...
loss with respect to its local batch $b$. Losses can then simply be summed across all devices (sum $A$). Individual collective permutes (for sum $B$) are fast (and indeed $D$ collective permutes is typically faster than two all-gathers between $D$ devices), and the memory cost at any given moment is reduced from $|B|^2$ to $b^2$ (for sum $C$). Usually $b$ is constant as scaling $|B|$ is achieved by increasing the number of accelerators. Due to being quadratic with respect to the batch size, the vanilla loss computation rapidly bottlenecks scaling up. This chunked approach enabled training with batch sizes over 1 million on relatively few devices.

4. Results

In this section, we evaluate the proposed SigLiT and SigLIP models across a wide range of batch sizes. We discuss what can be achieved with a small number of accelerator chips, using both SigLiT and SigLIP recipes. We also briefly discuss the impact of batch size on multilingual language image pre-training. We ablate the importance of our large-batch stabilization modification and the introduced learned bias term and present a study on the effect of positive and negative pairs ratio in the sigmoid loss. Lastly, we explore SigLIP’s data noise robustness.

To validate our models, we report zero-shot transfer results on the ImageNet dataset [14] and zero-shot retrieval results across 36 languages on the XM3600 dataset [41]. We use the ScalingViT-Adafactor optimizer [54] by default for all our experiments.

4.1. SigLiT: Scaling batch size to the limit

Following [55], we use the same precomputed embeddings for the images using a ViT-g vision model, and train a base size text tower from scratch with the same hyperparameters using the LiT image-text dataset [55]. We perform a study over a wide range of batch sizes, from 512 to 1 M, demonstrating the impact of batch size for contrastive learning. Results are presented in Figure 2 (left). When the batch size is smaller than 16 $k$, sigmoid loss outperforms softmax loss by a large margin. With growing batch sizes, we observe that softmax loss quickly catches up and potentially slightly underperforms sigmoid loss with a large enough batch size. Overall, we recommend using the SigLIP recipe for large batch sizes as well, due to the simplicity, compute savings, and straightforward memory efficient implementation.

There is a consensus that contrastive learning benefits from large batch sizes, while most of the existing studies stop at 64 $k$ batch size [55, 33, 10]. We successfully trained an SigLiT model at one million batch size, to explore the limit of contrastive learning. To our surprise, the performance saturates at 32 $k$ batch size, further scaling up the batch size only gives a minor boost, and the model peaks at 256 $k$ batch size. Our best SigLiT with a $B$ sized text mode achieves 84.7% zero-shot transfer accuracy on ImageNet, while the original LiT paper reports a slightly better 85.2% score with a 10 times larger $g$-sized text model. Figure 3 presents the impact of training duration for different batch sizes. It demonstrates that large, 262 $k$ batch size signifi-
Figure 3: SigLiT ImageNet 0-shot transfer results with different training durations. Large batch size results in a big performance boost, but needs a sufficiently long schedule to ramp up, as for short schedules, very large batch size results in a small number of gradient update steps.

4.2. SigLIP: Sigmoid loss is beneficial for language-image pre-training

We pre-train SigLIP models on the WebLI dataset [13], using only English image and text pairs. We use moderately-sized models: B/16 ViT for image embeddings and B-sized transformer for text embeddings. The input images are resized to $224 \times 224$ resolution. The text is tokenized by a 32 k vocabulary sentencepiece tokenizer [27] trained on the English C4 dataset [35], and a maximum of 16 text tokens are kept. Figure 2 middle plot shows SigLIP results. With less than 32 k batch size, SigLIP outperforms CLIP baselines with the standard softmax loss. On the other end of the scale, the memory efficiency of the sigmoid loss enabled much larger batch sizes. On the other end of the scale, the memory efficiency of the sigmoid loss enabled much larger batch sizes. For example, with four TPU-v4 chips, we could fit a batch size of 4096 with a Base SigLIP but only 2048 with a corresponding CLIP model. The two advantages together demonstrate significant benefits of the sigmoid loss for language image pre-training with fixed resources, which will be discussed in Section 4.5.

As batch size increases, the gap between the sigmoid and the softmax losses diminish. SigLIP performs best at batch size 32 k, whereas the softmax loss required 98 k for optimal performance and still didn’t outperform the sigmoid based variant. Scaling further, a larger batch size like 307 k hurts both losses.

4.3. mSigLIP: Multi-lingual pre-training

We further scale up the training data by keeping all the 100 languages from the WebLI dataset [13]. With multilingual data, one usually needs to use a larger international vocabulary. We first verify the impact of two tokenizers: a small multilingual vocabulary with 32 k tokens [35], and a large multilingual vocabulary with 250 k tokens [50]. We train B-sized ViT and text models for 900 M total examples seen, and observe slightly more than 1% improvement when using a larger vocabulary.

However, the token embeddings become huge for very large vocabulary sizes. Following the standard setup, we would need to store a $N \times W$ token embedding lookup table to train the multilingual model, where $N$ is the vocabulary size mentioned above and $W$ is the embedding dimension of the text model. To save memory, we propose to use a “bottlenecked” token embedding. We use $N \times K$ embedding matrix and additional $K \times W$ projection, where the bottleneck $K$ is much smaller than $W$.

In our experiments, we observed that using a large multilingual vocabulary with a bottleneck can be scaled up as efficiently as using a small multilingual vocabulary. Specifically, by enabling the bottleneck of size $K = 96$ for Base architecture with $W = 768$, we only see about a half percent quality drop on ImageNet zero-shot transfer, compared to using the full 250k vocabulary.

With the memory improvements, we train mSigLIP models for various batch sizes, for a total of 30 billion examples seen. Table 2 and Figure 2 (right plot) show the results. We were expecting a large batch size to improve multilingual pre-training, where the model sees more examples from the same language as hard negatives in a sin-

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>INet-0</th>
<th>XM Avg</th>
<th>XM De</th>
<th>XM En</th>
<th>XM Hi</th>
<th>XM Ru</th>
<th>XM Zh</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 k</td>
<td>71.6</td>
<td>34.8</td>
<td>54.7</td>
<td>46.5</td>
<td>9.1</td>
<td>50.1</td>
<td>30.7</td>
</tr>
<tr>
<td>32 k</td>
<td>73.2</td>
<td>34.9</td>
<td>54.8</td>
<td>46.2</td>
<td>8.5</td>
<td>49.9</td>
<td>32.5</td>
</tr>
<tr>
<td>64 k</td>
<td>73.2</td>
<td>34.4</td>
<td>55.4</td>
<td>46.5</td>
<td>7.9</td>
<td>49.7</td>
<td>32.0</td>
</tr>
<tr>
<td>128 k</td>
<td>73.2</td>
<td>33.6</td>
<td>54.3</td>
<td>46.6</td>
<td>8.1</td>
<td>48.6</td>
<td>30.6</td>
</tr>
<tr>
<td>240 k</td>
<td>73.1</td>
<td>32.7</td>
<td>54.7</td>
<td>46.6</td>
<td>7.3</td>
<td>49.3</td>
<td>23.7</td>
</tr>
</tbody>
</table>

Table 2: Multilingual SigLIP results with various batch sizes, pre-trained for 30 billion seen examples. We report zero-shot transfer results on ImageNet (INet-0) and averaged text to image retrieval results across 36 languages on the crossmodal 3600 dataset (XM). The full table on 36 languages can be found in Appendix.
Figure 4: Top: SigLIP with pre-trained encoders ramps up quickly. However, only disabling weight decay on the pre-trained encoder weights leads to stable behavior and good ImageNet 0-shot transfer results. Bottom: ImageNet 10-shot transfer results, where decaying the pre-trained weights leads to deterioration of the pre-trained model representation quality. Disabling weight decay makes the curve flatter.

gle mini-batch. However, we didn’t observe clear improvements with a batch size larger than 32 k. A batch size of 32 k is sufficient for a multilingual setup as well. On the XM3600 cross-modal retrieval tasks, we found that going beyond 32 k batch size leads to worse results on average while on ImageNet zero-shot transfer it stays flat. mSigLIP sets the new state-of-the-art on XM3600 text to image retrieval task, with only a Base size model. Our best result is 34.9%, which is more than 6% higher than the previously reported result 28.5% [13] with a standard LiT model [55] using a much larger four billion ViT-e model.

4.4. SigLiT with four TPU-v4 chips

For many practitioners, the important question usually is “what can be trained with a limited amount of resources?” We explore the usage of SigLiT models in this section with only four TPU-v4 chips, as the memory efficient sigmoid loss is suitable for this application scenario.

We follow the same setup as in section 4.1. We use the publicly available ViT-Augreg-B/8 [39] model as the frozen ($) vision tower, and precompute embeddings to accelerate the training [55]. The text model is a Large Transformer, but with a depth of only 12 layers (instead of 24). It is trained using the LION [12] optimizer with decoupled weight decay $1 \times 10^{-7}$, linearly warm-up of learning rate over 6.5k steps up to a peak of $1 \times 10^{-4}$, followed by a cosine decay to 0. We train for a total of 65,000 steps with a batch size of 32k – this leads to just under one day of training. Table 1 shows the results when training a model on four chips for one day, achieving 79.7% 0-shot ImageNet classification accuracy; very competitive in this limited resource regime. With a ViT-g/14 [54] model as the vision tower and a Large text tower, we can train at 20 k batch size on four chips for 107 k steps in under two days. This further pushes the 0-shot ImageNet classification accuracy up to 84.5%.

4.5. SigLIP with a small amount of TPU-v4 chips

It’s resource demanding to train a CLIP model from-scratch in general, with SigLIP it’s possible to fit a larger train batch size with fewer amount of chips. In this section, we explore ways to train SigLIP models efficiently with pre-trained weights. We use pre-trained weights to initialize the image model to accelerate the pre-training, which was originally discussed in [55]. We use the public and unlocked ($) ViT-Augreg-B/16 [39] model to initialize our vision tower and fine-tune on the same WebLI English data as used for SigLIP. In all the experiments, we apply a 0.1 learning rate multiplier to the pre-trained image tower to make it suitable
Figure 6: The effect of batch composition. We simulate various batch compositions by masking out negatives, either randomly, keeping only the hardest, or the easiest. With no masking, we have 16k negatives for each positive in the batch (1:16k) and the strongest masking we apply (1:1.6) results in almost balanced minibatches. In one setting we match total pairs seen by training for significantly longer. We observe ImageNet 0-shot score, the final value of the learned bias, and the average logits of positive and negative pairs. Overall, the imbalance does not seem to be detrimental, but finding an efficient way of mining negatives might be beneficial.

Figure 4 presents unlocked fine-tuning results alongside from-scratch randomly initialized baselines. We used 16 TPU-v4 chips and train at 16k batch size for 2.4B examples seen. We found that the fine-tuning setup doesn’t perform well out-of-the-box; this is consistent with prior works [55] where finetuning image models degraded visual representation quality. This is evidenced by ImageNet 10-shot linear classification, where in Figure 4 the fine-tuned setup is barely better than the from-scratch baseline.

We hypothesize that the default weight decay applied to the pre-trained weights reduces their effectiveness. Motivated by the fine-tuning recipe from [17, 54, 25], that uses no weight decay, we also propose disabling weight decay on the pre-trained weights for SigLIP training. Weight decay is therefore only applied to the randomly initialized weights in the text model. This simple modification significantly improved SigLIP results. Figure 4 shows that with our improved recipe, SigLIP reaches 71% 0-shot accuracy on ImageNet, using 16k batch size, trained on 16 chips for three days. We also present from-scratch results in the bottom rows of Table 1: with 32 TPUv4 chips for only two days, SigLIP achieves 72.1% 0-shot accuracy. This presents a significant training cost reduction e.g. compared to CLIP (approx. 2500 TPUv3-days for 72.6%) reported in [29].

4.6. Scaling up SigLIP
In this section, we scale up SigLIP by “overtraining” the model [42, 1] for a long schedule. We present results in Table 3 using ViT-B or ViT-L as the vision encoder, while the text encoder is kept the same size (B or L respectively). Following the recipe as described in Section 4.2, we train both models for 40 billion examples seen, but increase the number of image patches from \((224/16)^2 = 196\) to \((256/16)^2 = 256\) and the number of text tokens from 16 to 64. To get SigLIP results for different resolutions, we continue training the pre-trained model for 5 billion examples on the target resolutions (i.e. 224 or 384), using a 100x smaller learning rate without weight decay. In Table 3, we report zero-shot classification results on ImageNet [14], ObjectNet [2], ImageNetv2 [36], ImageNet ReaL [3], and zero-shot image-to-text (I\(\rightarrow\)T) retrieval, text-to-image (I\(\rightarrow\)T) retrieval results on MSCOCO [11]. Our SigLIP models outperform the previous models consistently with ViT-B and ViT-L architectures.

4.7. Stabilizing large-batch training
As we move to large batch sizes, the language image pretraining using transformers becomes increasingly more unstable, even when using a modestly-sized model (e.g. Base size). The reason for these instabilities is large spikes in the gradient norms, which translate to large-magnitude changes in the weights that may destabilize the training process, see Figure 5. We observe that reducing \(\beta_2\) in Adam and AdaFactor from its default 0.999 to 0.95 (which was suggested in [20, 9]) is enough to stabilize the training. Intuitively, this allows recovering from gradient spikes quicker. We opt for setting \(\beta_2 = 0.95\) for all our experiments.

4.8. Negative ratio in sigmoid loss
One question which arises when shifting the perspective from the softmax’s “pick the right class” view to the sig-
Table 3: **Comparison with other methods.** Our SigLIP models outperform the previous models, e.g. OpenCLIP [22] and CLIP [34], by a significant margin on both the zero-shot classification tasks and the zero-shot retrieval tasks. Compared to the recently introduced EVA-CLIP [40] model, SigLIP-L outperforms EVA-CLIP-L on all the tasks with the same number of 256 patches. SigLIP-L (256 patches) also outperforms EVA-CLIP-L (576 patches) by a large margin on the retrieval tasks.

Table 4: **Bias (b) and temperature (t’) initialization.** Results are reported using Base architecture, 8 k batch size, trained for 900M examples. Enabling the bias term b with −10 initialization improves results consistently.
4.9. Bias term in sigmoid loss

We ablate the bias term in the loss function, using the Base architecture with an 8 k batch size, trained for 900M examples with the SigLIP setup. Zero-shot transfer results are reported on ImageNet [14], Oxford-iiit pet [32] and Cifar100 [26]. Table 4 presents results with and without a bias term in the sigmoid loss.

Enabling the bias term with a $-10$ initialization consistently improves performance across all tasks. This is because the bias term ensures that the training starts close to the prior, preventing dramatic over-correction in early optimization. In contrast, a randomly chosen bias term initialization, such as the 0 initialization in Table 4, fails to address the over-correction issue, leading to significantly worse results. This effect is particularly noticeable when using a small temperature $t’$ initialization. We set the bias and temperature initialization to $b = -10$ and $t’ = \log 10$ (hence $t = 10$) as the default for all experiments.

4.10. Label noise robustness

Prior works demonstrated improved robustness against label noise when using the sigmoid loss for classification models [3]. This property would be particularly useful here in the face of the famously noisy nature of popular large-scale image-text datasets. In order to study this for SigLIP, we train M/16 image models alongside an M text model at batch size 16384 for 3.6 billion seen examples. We corrupt the training data using one of the following methods:

- **Image**: With probability $p$, replace the image with uniform random noise.
- **Text**: With probability $p$, replace tokenized text with a new sequence of randomly sampled tokens, up to some (sampled) sequence length.
- **Batch alignment**: Randomly shuffle the ordering of $p\%$ of the batch.
- **Image & text**: Apply both with probability $p$ each.
- **Image, text & batch**: Alongside (4), also shuffle fraction $p$ of alignments.

Results from varying the likelihood of the corruption are shown in Figure 7. Models trained with sigmoid loss are increasingly robust to all kinds of added noise.

5. Conclusion

We conducted a study on two language-image pre-training instances that used the sigmoid loss: SigLit and SigLIP. Our results demonstrate that the sigmoid loss performs better than the softmax baseline, particularly for small train batch sizes. This loss function is also more memory efficient, which allows larger train batch sizes without requiring additional resources. We performed a thorough investigation of the batch size in contrastive learning. Surprisingly, we found that a relatively modest batch size of 32 k yielded nearly optimal performance. Further studies have been performed to understand better the introduced bias term in the sigmoid loss, robustness to data noises and the impact of positive and negative pairs ratio in the sigmoid loss. We hope this work will facilitate language-image pre-training research with limited resources.

Acknowledgements. We thank Daniel Keysers, Ilya Tolstikhin, Olivier Bousquet and Michael Tschannen for their valuable feedback and discussions on this paper. We thank Joan Puigcerver, Josip Djolonga and Black Hechtman for discussions on efficient implementations of the chunked contrastive loss. We thank Kaiming He and Xinlei Chen for the discussion of $\beta_2$ to stabilize the training. We also thank Ross Wightman for spotting a mistake in the pseudocode in the first version of this paper. Similarly, we thank Boris Dayma for spotting a typo making $t$ vs $t’$ confusing, which we fixed in the third version of this paper. We thank the Google Deepmind team for providing a supportive research environment. We use the big vision codebase [5, 4] for all experiments in this project.
References


[23] Chao Jia, Yinxfei Yang, Ye Xia, Yi-Ting Chen, Zaraana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsun Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In ICML, 2021.


[40] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. EVA-CLIP: improved training techniques for CLIP at scale. *CoRR*, abs/2303.15389, 2023. 8


